ПРИЛОЖЕНИЕ Д

ТЕКСТ ПРОГРАММЫ.

АННОТАЦИЯ

Данный документ предназначен для описания объектов базы данных и модулей программы, а также представление скрипта БД и машинного кода программы.

Содержание

[1. ОСНОВНАЯ ЧАСТЬ ТЕКСТА ПРОГРАММЫ 3](#_Toc40548440)

[1.1 Описание структуры базы данных 3](#_Toc40548441)

[1.2 Скрипт DLL базы данных 9](#_Toc40548442)

[1.3 Описание структуры программы 17](#_Toc40548443)

[1.4 Код структуры программы 19](#_Toc40548444)

# 1. ОСНОВНАЯ ЧАСТЬ ТЕКСТА ПРОГРАММЫ

## 1.1 Описание структуры базы данных

Таблица 1.1 - описание объектов БД

|  |  |
| --- | --- |
| Название | Описание |
| Таблицы | |
| [dbo].[Authorization] | Таблица «Авторизация» предназначена для занесения в базу данных информации об авторизации, содержит в себе «Логин», «Пароль», «ID\_Role» . |
| [dbo].[Ammo] | Таблица «Патроны» предназначена для занесения в базу данных информации о патронах, содержит в себе «Тип патронов», «Количество», «Цена» . |
| [dbo].[Dogovor] | Таблица «Договор» предназначена для занесения в базу данных информации о договорах, содержит в себе «Номер договора». |
| [dbo].[ Employee] | Таблица «Сотрудники» предназначена для занесения в базу данных информации о сотрудниках, содержит в себе «Фамилия», «Имя», «Отчество», «Опыт работы», «Дата поступления на работу», «Position\_ID». |
| [dbo].[Klient] | Таблица «Клиенты» предназначена для занесения в базу данных информации о клиентах, содержит в себе «Фамилия», «Имя», «Отчество», «Телефонный номер», «License\_ID». |
| [dbo].[License] | Таблица «Лицензия» предназначена для занесения в базу данных информации о лицензиях, содержит в себе «Номер лицензии». |
| [dbo].[Modifications] | Таблица «Модификации» предназначена для занесения в базу данных информации о модификациях, содержит в себе «Название модификации», «Количество», «Цена» . |
| [dbo].[ Nakladnaya] | Таблица «Накладная» предназначена для занесения в базу данных информации о накладных, содержит в себе «Номер накладной», «Supply\_ID». |
| [dbo].[Position] | Таблица «Должность» предназначена для занесения в базу данных информации о должностях, содержит в себе «Название должности», «Оплата». |
| [dbo].[Role] | Таблица «Роль» предназначена для хранения информации о ролях, которые выдаются при регистрации, содержит в себе «Название роли», «Роль клиента», «Роль сотрудника», «Роль администратора» |
| [dbo].[ Supplier] | Таблица «Поставщики» предназначена для занесения в базу данных информации о поставщиках, содержит в себе «Название организации-поставщика». |
| [dbo].[Supply] | Таблица «Поставки» предназначена для занесения в базу данных информации о поставках товара в магазин, содержит в себе «Дата», «Кол-во принятых оружий», «Кол-во принятых модификаций», «Кол-во принятых патронов», «Supplier\_ID», «Dogovor\_ID». |
| [dbo].[ Type\_Weapon] | Таблица «Тип оружия» предназначена для занесения в базу данных информации о тпах оружия, содержит в себе «Названиетипа оружия». |
| [dbo].[Weapon] | Таблица «Оружие» предназначена для занесения в базу данных информации о продаваемом оружии, содержит в себе «Название оружия», «Точность», «Скорострельность», «Кол-во патронов в магазине», «Кол-во оружия на складе», «Цена», «Type\_Weapon\_ID». |
| Представления | |
| [dbo].[ Klient\_select\_view] | Представление «Информация о клиенте» предназначено для хранения информации о клиенте, содержит ФИО клиента, номер лицензии. |
| [dbo].[ Weapon\_select\_view] | Представление «Информация об оружии» предназначено для хранения информации об оружии, содержит информацию об оружии+его тип. |
| Хранимые процедуры | |
| [dbo].[Ammo\_insert] | Хранимая процедура, предназначенная для осуществления процесса добавления новой записи в таблицу «Патроны» |
| [dbo].[ Ammo\_update] | Хранимая процедура, предназначенная для осуществления процесса изменения уже существующей записи в таблице «Патроны» |
| [dbo].[Ammo\_delete] | Хранимая процедура, предназначенная для осуществления процесса удаления существующей записи в таблице «Патроны» |
| [dbo].[ Ammo\_select] | Хранимая процедура, предназначенная для осуществления процесса выведения значений из таблицы «Патроны» |
| [dbo].[Klient\_select\_detail] | Хранимая процедура, предназначенная для осуществления процесса выведения значений из представления «Информация о клиенте» |
| [dbo].[Weapon\_select\_detail] | Хранимая процедура, предназначенная для осуществления процесса выведения значений из представления «Информация об оружии» |

Для каждой из вышеперечисленныъ таблиц в БД прописаны аналогичные процедуры по 4 на каждую таблице. На 2 представления отводится по одной процедуре select\_detail. => Всего процедур 14 (кол-во таблиц)\*4 (кол-во одинаковых процедур на каждую таблицу) = 56 аналогичных процедур. Подсчеты и этот вывод сделаны для экономии времени как моего, так и вашего.

## 1.2 Скрипт DLL базы данных

--drop database [Weapon\_Store]

--go

set ansi\_padding on

go

set quoted\_identifier on

go

set ansi\_nulls on

go

create database [Weapon\_Store]

go

use [Weapon\_Store]

go

create table [dbo].[Authorization]

(

[ID\_Authorization] [int] not null identity(1,1),

[Login] [varchar] (30) not null,

[Password] [varchar] (30) not null,

[ID\_Role][int] not null

constraint [PK\_Authorization] primary key clustered ([ID\_Authorization] ASC) on [PRIMARY],

constraint [UQ\_Login] unique ([Login]),

constraint [CH\_Password\_Number] check ([Password] like '%[0-9]%'),

constraint [CH\_Password\_Latter] check ([Password] like '%[A-Z]%'),

constraint [CH\_Password\_Symbole] check ([Password] like '%[!@#$%^&\*()\_+=]%'),

constraint [CH\_Login] check (len([Login]) >= 8),

constraint [CH\_Password] check (len([Password]) >= 8)

)

go

create procedure [dbo].[Authorization\_select]

as

select \* from [dbo].[Authorization]

go

insert into [dbo].[Authorization] ([Login], [Password])

values ('Ivanov\_P\_S','QwertY\_01'),

('Olegova\_E\_E','QwertY\_02'),

('Petrov\_O\_P','QwertY\_03'),

('Mihailova\_O\_N','QwertY\_04'),

('Dmitrov\_A\_A','QwertY\_05'),

('Zaharova\_A\_R','QwertY\_06'),

('Pavlov\_E\_A','QwertY\_07'),

('Egorov\_D\_K','QwertY\_08'),

('Vasinkov\_N\_A','QwertY\_09'),

('Sergeev\_A\_S','QwertY\_10'),

('Rinatova\_J\_K','QwertY\_11'),

('Andreeva\_L\_A','QwertY\_12')

go

create procedure [dbo].[Authorization\_update]

@ID\_Authorization [int], @Login [varchar] (30), @Password [varchar] (30),@ID\_Role[int]

as

update [dbo].[Authorization] set

[Login] = @Login,

[Password] = @Password,

[ID\_Role] = @ID\_Role

where

[ID\_Authorization] = @ID\_Authorization

go

create procedure [dbo].[Authorization\_insert]

@Login [varchar] (30), @Password [varchar] (30),@ID\_Role[int]

as

insert into [dbo].[Authorization] ([Login], [Password],[ID\_Role])

values (@Login, @Password,@ID\_Role)

go

create procedure [dbo].[Authorization\_delete]

@ID\_Authorization [int]

as

delete from [dbo].[Authorization]

where

[ID\_Authorization] = @ID\_Authorization

go

create table [dbo].[License]

(

[ID\_License] [int] identity(1,1) not null,

[License\_Number] [varchar] (30) not null,

constraint [PK\_License] primary key clustered ([ID\_License] ASC) on [PRIMARY],

)

go

create procedure [dbo].[License\_select]

as

select \* from [dbo].[License]

go

create procedure [dbo].[License\_update]

@ID\_License [int], @License\_Number [varchar] (30)

as

update [dbo].[License] set

[License\_Number] = @License\_Number

where

[ID\_License] = @ID\_License

go

create procedure [dbo].[License\_insert]

@License\_Number [varchar] (30)

as

insert into [dbo].[License] ([License\_Number])

values (@License\_Number)

go

create procedure [dbo].[License\_delete]

@ID\_License [int]

as

delete from [dbo].[License]

where

[ID\_License] = @ID\_License

go

create table [dbo].[Klient]

(

[ID\_Authorization] [int] not null,

[First\_Name\_Klient] [varchar] (30) not null,

[Name\_Klient] [varchar] (30) not null,

[Middle\_Name\_Klient] [varchar] (30) null default('No data'),

[Phone\_Number] [varchar] (16) not null,

[License\_ID] [int] not null,

constraint [FK\_Klient] foreign key ([ID\_Authorization]) references [dbo].[Authorization] ([ID\_Authorization]),

constraint [PK\_Klient] primary key clustered ([ID\_Authorization] ASC) on [PRIMARY],

constraint [UQ\_Phone\_Number] unique ([Phone\_Number]),

constraint [FK\_License\_Klient] foreign key ([License\_ID]) references [dbo].[License] ([ID\_License])

)

go

create procedure [dbo].[Klient\_select]

as

select \* from [dbo].[Klient]

go

insert into [dbo].[Klient] ([First\_Name\_Klient], [Name\_Klient], [Middle\_Name\_Klient], [Phone\_Number],[License\_ID],[ID\_Authorization])

values ('Иванов','Пётр','Сергеевич','8-999-999-99-99','1'),

('Олегова','Екатерина','Евгеньевна','8-888-888-88-88','2'),

('Петров','Олег','Павлович','8-777-777-77-77','3'),

('Михайлова','Ольга','Николаевна','8-666-666-66-66','4'),

('Дмитров','Андрей','Александрович','8-555-555-55-55','5'),

('Захарова','Анастасия','Руслановна','8-444-444-44-44','6')

go

create procedure [dbo].[Klient\_update]

@ID\_Authorization [int], @First\_Name\_Klient [varchar] (30), @Name\_Klient [varchar] (30), @Middle\_Name\_Klient [varchar] (30), @Phone\_Number [varchar] (16), @License\_ID [int]

as

update [dbo].[Klient] set

[First\_Name\_Klient] = @First\_Name\_Klient,

[Name\_Klient] = @Name\_Klient,

[Middle\_Name\_Klient] = @Middle\_Name\_Klient,

[Phone\_Number] = @Phone\_Number,

[License\_ID] = @License\_ID

where

[ID\_Authorization] = @ID\_Authorization

go

create procedure [dbo].[Klient\_insert]

@ID\_Authorization [int], @First\_Name\_Klient [varchar] (30), @Name\_Klient [varchar] (30), @Middle\_Name\_Klient [varchar] (30), @Phone\_Number [char] (19), @License\_ID [int]

as

insert into [dbo].[Klient] ([ID\_Authorization],[First\_Name\_Klient], [Name\_Klient], [Middle\_Name\_Klient], [Phone\_Number], [License\_ID])

values (@ID\_Authorization, @First\_Name\_Klient, @Name\_Klient, @Middle\_Name\_Klient, @Phone\_Number, @License\_ID)

go

create procedure [dbo].[Klient\_delete]

@ID\_Authorization [int]

as

delete from [dbo].[Klient]

where

[ID\_Authorization] = @ID\_Authorization

go

create table [dbo].[Position]

(

[ID\_Position] [int] identity(1,1) not null,

[Name\_Position] [varchar] (20) not null,

[Salary] [varchar] (20) not null,

constraint [PK\_Position] primary key clustered ([ID\_Position] ASC) on [PRIMARY],

)

go

create procedure [dbo].[Position\_select]

as

select \* from [dbo].[Position]

go

create procedure [dbo].[Position\_update]

@ID\_Position [int], @Name\_Position [varchar] (20), @Salary [varchar] (20)

as

update [dbo].[Position] set

[Name\_Position] = @Name\_Position,

[Salary] = @Salary

where

[ID\_Position] = @ID\_Position

go

create procedure [dbo].[Position\_insert]

@Name\_Position [varchar] (20), @Salary [varchar] (20)

as

insert into [dbo].[Position] ([Name\_Position], [Salary])

values (@Name\_Position, @Salary)

go

create procedure [dbo].[Position\_delete]

@ID\_Position [int]

as

delete from [dbo].[Position]

where

[ID\_Position] = @ID\_Position

go

create table [dbo].[Employee]

(

[ID\_Authorization] [int] not null,

[First\_Name\_Employee] [varchar] (30) not null,

[Name\_Employee] [varchar] (30) not null,

[Middle\_Name\_Employee] [varchar] (30) null default('No data'),

[Job\_Experience] [varchar] (10) not null,

[Employment\_Data] [varchar](10) not null,

[Position\_ID] [int] not null,

constraint [PK\_Employee] primary key clustered ([ID\_Authorization] ASC) on [PRIMARY],

constraint [FK\_Employee] foreign key ([ID\_Authorization]) references [dbo].[Authorization] ([ID\_Authorization]),

constraint [FK\_Position\_Employee] foreign key ([Position\_ID])

references [dbo].[Position] ([ID\_Position])

)

go

create procedure [dbo].[Employee\_select]

as

select \* from [dbo].[Employee]

go

create procedure [dbo].[Employee\_update]

@ID\_Authorization [int], @First\_Name\_Employee [varchar] (30), @Name\_Employee [varchar] (30), @Middle\_Name\_Employee [varchar] (30), @Job\_Experience [varchar] (10), @Employment\_Data [varchar] (10), @Position\_ID [int]

as

update [dbo].[Employee] set

[First\_Name\_Employee] = @First\_Name\_Employee,

[Name\_Employee] = @Name\_Employee,

[Middle\_Name\_Employee] = @Middle\_Name\_Employee,

[Job\_Experience] = @Job\_Experience,

[Employment\_Data] = @Employment\_Data,

[Position\_ID] = @Position\_ID

where

[ID\_Authorization] = @ID\_Authorization

go

create procedure [dbo].[Employee\_insert]

@ID\_Authorization [int],@First\_Name\_Employee [varchar] (30), @Name\_Employee [varchar] (30), @Middle\_Name\_Employee [varchar] (30), @Job\_Experience [varchar] (10), @Employment\_Data [varchar] (10), @Position\_ID [int]

as

insert into [dbo].[Employee] ([ID\_Authorization],[First\_Name\_Employee], [Name\_Employee], [Middle\_Name\_Employee], [Job\_Experience], [Employment\_Data], [Position\_ID])

values (@ID\_Authorization,@First\_Name\_Employee, @Name\_Employee, @Middle\_Name\_Employee, @Job\_Experience, @Employment\_Data, @Position\_ID)

go

create procedure [dbo].[Employee\_delete]

@ID\_Authorization [int]

as

delete from [dbo].[Employee]

where

[ID\_Authorization] = @ID\_Authorization

go

create table [dbo].[Type\_Weapon]

(

[ID\_Type\_Weapon] [int] identity(1,1) not null,

[Name\_Type\_Weapon] [varchar] (30) not null,

constraint [PK\_Type\_Weapon] primary key clustered ([ID\_Type\_Weapon] ASC) on [PRIMARY],

)

go

create procedure [dbo].[Type\_Weapon\_select]

as

select \* from [dbo].[Type\_Weapon]

go

create procedure [dbo].[Type\_Weapon\_update]

@ID\_Type\_Weapon [int], @Name\_Type\_Weapon [varchar] (30)

as

update [dbo].[Type\_Weapon] set

[Name\_Type\_Weapon] = @Name\_Type\_Weapon

where

[ID\_Type\_Weapon] = @ID\_Type\_Weapon

go

create procedure [dbo].[Type\_Weapon\_insert]

@Name\_Type\_Weapon [varchar] (30)

as

insert into [dbo].[Type\_Weapon] ([Name\_Type\_Weapon])

values (@Name\_Type\_Weapon)

go

create procedure [dbo].[Type\_Weapon\_delete]

@ID\_Type\_Weapon [int]

as

delete from [dbo].[Type\_Weapon]

where

[ID\_Type\_Weapon] = @ID\_Type\_Weapon

go

create table [dbo].[Weapon]

(

[ID\_Weapon] [int] not null identity(1,1),

[Name\_Weapon] [varchar] (30) not null,

[Accuracy] [varchar] (30) not null,

[Fire\_Rate] [varchar] (30) not null,

[Shells\_In\_Store] [varchar] (30) not null,

[Ammount\_Weapon] [int] not null,

[Cost] [varchar] (20) not null,

[Type\_Weapon\_ID] [int] not null,

constraint [PK\_Weapon] primary key clustered ([ID\_Weapon] ASC) on [PRIMARY],

constraint [FK\_Type\_Weapon\_Weapon] foreign key ([Type\_Weapon\_ID])

references [dbo].[Type\_Weapon] ([ID\_Type\_Weapon]),

)

go

create procedure [dbo].[Weapon\_select]

as

select \* from [dbo].[Weapon]

go

create procedure [dbo].[Weapon\_update]

@ID\_Weapon [int],@Name\_Weapon [varchar] (30),@Accuracy [varchar] (30),@Fire\_Rate [varchar] (30),@Shells\_In\_Store [varchar] (30),@Ammount\_Weapon [int],@Cost [varchar] (20),@Type\_Weapon\_ID [int]

as

update [dbo].[Weapon] set

[Name\_Weapon] = @Name\_Weapon,

[Accuracy]=@Accuracy,

[Fire\_Rate]=@Fire\_Rate,

[Shells\_In\_Store]=@Shells\_In\_Store,

[Ammount\_Weapon]=@Ammount\_Weapon,

[Cost]=@Cost,

[Type\_Weapon\_ID]=@Type\_Weapon\_ID

where

[ID\_Weapon] = @ID\_Weapon

go

create procedure [dbo].[Weapon\_insert]

@Name\_Weapon [varchar] (30),@Accuracy [varchar] (30),@Fire\_Rate [varchar] (30),@Shells\_In\_Store [varchar] (30),@Ammount\_Weapon [int],@Cost [varchar] (20),@Type\_Weapon\_ID [int]

as

insert into [dbo].[Weapon] ([Name\_Weapon],[Accuracy],[Fire\_Rate],[Shells\_In\_Store],[Ammount\_Weapon],[Cost],[Type\_Weapon\_ID])

values (@Name\_Weapon,@Accuracy,@Fire\_Rate,@Shells\_In\_Store,@Ammount\_Weapon, @Cost,@Type\_Weapon\_ID)

go

create procedure [dbo].[Weapon\_delete]

@ID\_Weapon [int]

as

delete from [dbo].[Weapon]

where

[ID\_Weapon] = @ID\_Weapon

go

create view [dbo].[Weapon\_select\_view]

as

select ID\_Weapon, ([Name\_Weapon] +' '+ [Name\_Type\_Weapon]) as WeaponInfo, Accuracy, Fire\_Rate,Shells\_In\_Store, Ammount\_Weapon,Cost from Weapon

inner join Type\_Weapon on Type\_Weapon.ID\_Type\_Weapon = Weapon.Type\_Weapon\_ID

go

create procedure [dbo].[Weapon\_select\_detail]

as

select \* from [dbo].[Weapon\_select\_view]

go

create table [dbo].[Supplier]

(

[ID\_Supplier] [int] identity(1,1) not null,

[Name\_Organization] [varchar] (30) not null,

constraint [PK\_Supplier] primary key clustered ([ID\_Supplier] ASC) on [PRIMARY],

)

go

create procedure [dbo].[Supplier\_select]

as

select \* from [dbo].[Supplier]

go

create procedure [dbo].[Supplier\_update]

@ID\_Supplier [int], @Name\_Organization [varchar] (30)

as

update [dbo].[Supplier] set

[Name\_Organization] = @Name\_Organization

where

[ID\_Supplier] = @ID\_Supplier

go

create procedure [dbo].[Supplier\_insert]

@Name\_Organization [varchar] (30)

as

insert into [dbo].[Supplier] ([Name\_Organization])

values (@Name\_Organization)

go

create procedure [dbo].[Supplier\_delete]

@ID\_Supplier [int]

as

delete from [dbo].[Supplier]

where

[ID\_Supplier] = @ID\_Supplier

go

create table [dbo].[Dogovor]

(

[ID\_Dogovor] [int] identity(1,1) not null,

[Number\_Dogovor] [varchar] (30) not null,

constraint [PK\_Dogovor] primary key clustered ([ID\_Dogovor] ASC) on [PRIMARY],

)

go

create procedure [dbo].[Dogovor\_select]

as

select \* from [dbo].[Dogovor]

go

create procedure [dbo].[Dogovor\_update]

@ID\_Dogovor [int], @Number\_Dogovor [varchar] (30)

as

update [dbo].[Dogovor] set

[Number\_Dogovor] = @Number\_Dogovor

where

[ID\_Dogovor] = @ID\_Dogovor

go

create procedure [dbo].[Dogovor\_insert]

@Number\_Dogovor [varchar] (30)

as

insert into [dbo].[Dogovor] ([Number\_Dogovor])

values (@Number\_Dogovor)

go

create procedure [dbo].[Dogovor\_delete]

@ID\_Dogovor [int]

as

delete from [dbo].[Dogovor]

where

[ID\_Dogovor] = @ID\_Dogovor

go

create table [dbo].[Supply]

(

[ID\_Supply] [int] identity(1,1) not null,

[Date] [varchar] (10) not null,

[Ammount\_Accepted\_Weapon][int] not null,

[Ammount\_Accepted\_Modifications][int] not null,

[Ammount\_Accepted\_Ammo][int] not null,

[Supplier\_ID] [int] not null,

[Dogovor\_ID] [int] not null,

constraint [PK\_Supply] primary key clustered ([ID\_Supply] ASC) on [PRIMARY],

constraint [FK\_Supplier\_Supply] foreign key ([Supplier\_ID])

references [dbo].[Supplier] ([ID\_Supplier]),

constraint [FK\_Dogovor\_Supply] foreign key ([Dogovor\_ID])

references [dbo].[Dogovor] ([ID\_Dogovor])

)

go

create procedure [dbo].[Supply\_select]

as

select \* from [dbo].[Supply]

go

create procedure [dbo].[Supply\_update]

@ID\_Supply [int], @Date [varchar] (10),@Ammount\_Accepted\_Weapon[int],@Ammount\_Accepted\_Modifications[int],@Ammount\_Accepted\_Ammo[int],@Supplier\_ID [int], @Dogovor\_ID [int]

as

update [dbo].[Supply] set

[Date]=@Date,

[Ammount\_Accepted\_Weapon]=@Ammount\_Accepted\_Weapon,

[Ammount\_Accepted\_Modifications]=@Ammount\_Accepted\_Modifications,

[Ammount\_Accepted\_Ammo]=@Ammount\_Accepted\_Ammo,

[Supplier\_ID]=@Supplier\_ID,

[Dogovor\_ID] =@Dogovor\_ID

where

[ID\_Supply] = @ID\_Supply

go

create procedure [dbo].[Supply\_insert]

@Date [varchar] (10),@Ammount\_Accepted\_Weapon[int],@Ammount\_Accepted\_Modifications[int],@Ammount\_Accepted\_Ammo[int],@Supplier\_ID [int], @Dogovor\_ID [int]

as

insert into [dbo].[Supply] ([Date],[Ammount\_Accepted\_Weapon],[Ammount\_Accepted\_Modifications],[Ammount\_Accepted\_Ammo],[Supplier\_ID],[Dogovor\_ID])

values (@Date,@Ammount\_Accepted\_Weapon,@Ammount\_Accepted\_Modifications,@Ammount\_Accepted\_Ammo,@Supplier\_ID,@Dogovor\_ID)

go

create procedure [dbo].[Supply\_delete]

@ID\_Supply [int]

as

delete from [dbo].[Supply]

where

[ID\_Supply] = @ID\_Supply

go

create table [dbo].[Nakladnaya]

(

[ID\_Nakladnaya] [int] identity(1,1) not null,

[Number\_Nakladnaya] [varchar] (30) not null,

[Supply\_ID] [int] not null,

constraint [PK\_Nakladnaya] primary key clustered ([ID\_Nakladnaya] ASC) on [PRIMARY],

constraint [FK\_Supply\_Nakladnaya] foreign key ([Supply\_ID])

references [dbo].[Supply] ([ID\_Supply])

)

go

create procedure [dbo].[Nakladnaya\_select]

as

select \* from [dbo].[Nakladnaya]

go

create procedure [dbo].[Nakladnaya\_update]

@ID\_Nakladnaya [int], @Number\_Nakladnaya [varchar] (30),@Supply\_ID [int]

as

update [dbo].[Nakladnaya] set

[Number\_Nakladnaya]=@Number\_Nakladnaya,

[Supply\_ID]= @Supply\_ID

where

[ID\_Nakladnaya] = @ID\_Nakladnaya

go

create procedure [dbo].[Nakladnaya\_insert]

@Number\_Nakladnaya [varchar] (30),@Supply\_ID [int]as

insert into [dbo].[Nakladnaya] ([Number\_Nakladnaya],[Supply\_ID])

values (@Number\_Nakladnaya,@Supply\_ID)

go

create procedure [dbo].[Nakladnaya\_delete]

@ID\_Nakladnaya [int]

as

delete from [dbo].[Nakladnaya]

where

[ID\_Nakladnaya] = @ID\_Nakladnaya

go

create table [dbo].[Ammo]

(

[ID\_Ammo] [int] not null identity(1,1),

[Type\_Ammo] [varchar] (30) not null,

[Ammount\_Ammo] [int] not null,

[Cost] [varchar] (30) not null,

constraint [PK\_Ammo] primary key clustered ([ID\_Ammo] ASC) on [PRIMARY]

)

go

create procedure [dbo].[Ammo\_select]

as

select \* from [dbo].[Ammo]

go

create procedure [dbo].[Ammo\_update]

@ID\_Ammo [int],@Type\_Ammo [varchar] (30), @Ammount\_Ammo [int], @Cost [varchar] (30)

as

update [dbo].[Ammo] set

[Type\_Ammo] = @Type\_Ammo,

[Ammount\_Ammo]=@Ammount\_Ammo,

[Cost] = @Cost

where

[ID\_Ammo] = @ID\_Ammo

go

create procedure [dbo].[Ammo\_insert]

@Type\_Ammo [varchar] (30), @Ammount\_Ammo [int], @Cost [varchar] (30)

as

insert into [dbo].[Ammo] ([Type\_Ammo],[Ammount\_Ammo],[Cost])

values (@Type\_Ammo,@Ammount\_Ammo,@Cost)

go

create procedure [dbo].[Ammo\_delete]

@ID\_Ammo [int]

as

delete from [dbo].[Ammo]

where

[ID\_Ammo] = @ID\_Ammo

go

create table [dbo].[Modifications]

(

[ID\_Modification] [int] not null identity(1,1),

[Name\_Modification] [varchar] (30) not null,

[Ammount\_Modifications] [int] not null,

[Cost] [varchar] (30) not null,

constraint [PK\_Modification] primary key clustered ([ID\_Modification] ASC) on [PRIMARY]

)

go

create procedure [dbo].[Modifications\_select]

as

select \* from [dbo].[Modifications]

go

create procedure [dbo].[Modifications\_update]

@ID\_Modification [int], @Name\_Modification [varchar] (30), @Ammount\_Modifications [int], @Cost [varchar] (30)

as

update [dbo].[Modifications] set

[Name\_Modification] = @Name\_Modification,

[Ammount\_Modifications]=@Ammount\_Modifications,

[Cost] = @Cost

where

[ID\_Modification] = @ID\_Modification

go

create procedure [dbo].[Modifications\_insert]

@Name\_Modification [varchar] (30), @Ammount\_Modifications [int], @Cost [varchar] (30)

as

insert into [dbo].[Modifications] ([Name\_Modification],[Ammount\_Modifications],[Cost])

values (@Name\_Modification,@Ammount\_Modifications,@Cost)

go

create procedure [dbo].[Modifications\_delete]

@ID\_Modification [int]

as

delete from [dbo].[Modifications]

where

[ID\_Modification] = @ID\_Modification

go

create table [dbo].[Role]

(

[ID\_Role] [int] not null identity(1,1),

[Title\_Role] [varchar] (30) not null,

[Klient] [int] not null,

[Employee] [int] not null,

[Admin] [int] not null,

constraint [PK\_Role] primary key clustered ([ID\_Role] ASC) on [PRIMARY]

)

go

insert into [dbo].[Role] ([Title\_Role] ,[Add\_client] ,[Add\_Weapon] ,[Add\_Ammo],[Add\_Modifications],[Goods\_purchase],[Lock\_client] ,[Accounting\_goods] )

values

('Клиент',0,0,0,0,1,0,0),

('Продавец-консультант',1,0,0,0,1,1,1),

('Кладовщик',0,0,0,1,0,0,1),

('Администратор',1,1,1,1,0,1,1)

go

create procedure [dbo].[Role\_select]

@ID\_ROLE [int]

as

IF(@ID\_ROLE IS NULL OR @ID\_ROLE = '')

select \* from [dbo].[Role]

ELSE

select \* from [dbo].[Role] where [ID\_Role] = @ID\_Role

go

create procedure [dbo].[Role\_update]

@ID\_Role [int], @Title\_Role [int], @Klient [int], @Employee [int], @Admin [int]

as

update [dbo].[Role] set

[Title\_Role] = @Title\_Role,

[Klient] = @Klient,

[Employee] = @Employee,

[Admin]=@Admin,

where

[ID\_Role] = @ID\_Role

go

create procedure [dbo].[Role\_insert]

@Title\_Role [int], @Title\_Role [int], @Klient [int], @Employee [int], @Admin [int]

as

insert into [dbo].[Role] ([Title\_Role], [Klient], [Employee],[Admin])

values (@Title\_Role, @Klient, @Employee, @Admin)

go

create procedure [dbo].[Role\_delete]

@ID\_Role [int]

as

delete from [dbo].[Role]

where

[ID\_Role] = @ID\_Role

go

create procedure [dbo].[Klient\_select\_detail]

as

select \* from [dbo].[klient\_select\_view]

go

create view [dbo].[Klient\_select\_view]

as

select ID\_Authorization, ([First\_Name\_Klient] +' '+ [Name\_Klient]+' '+ [Middle\_Name\_Klient]+' '+[Phone\_Number]) as KlientInfo, License.License\_Number as NumberLicense from Klient

inner join License on License.ID\_License = Klient.License\_ID

go

## 1.3 Описание структуры программы

Таблица 1.2 - описание структуры программы

|  |  |
| --- | --- |
| Zastavka.xaml.cs | На данной форме проигрывается заставка приложени. |
| Authorization.xaml.cs | На данной форме проводится авторизация клиента/сотрудника по ролям |
| PersonalAccount.xaml.cs | Форма предназначена для отображения возможностей авторизованного пользователя. |
| Registration.xaml.cs | Форма, на которой регистрируется клиент. |
| KlientInfo.xaml.cs | Форма с отображением информации для клиентов. |
| Spravochiki.xaml.cs | Форма, на которой администратор моджет выбрать нужный ему справочник для редактирования. |
| Weapons.xaml.cs | Справочник оружия для редактирования информации администратором таблицы «Оружие» и «Тип оружия» |
| Modifications.xaml.cs | Справочник модификаций для редактирования информации администратором таблицы «Модификации» |
| Ammo.xaml.cs | Справочник патронов для редактирования информации администратором таблицы «Патроны» |
| Klients.xaml.cs | Форма с таблицей клиентов, здесь хранятся все данные клиентов при регистрации |
| BlockKlient.xaml.cs | Форма для блокировки или разблокировки клиентов. |
| Nakladnaya.xaml.cs | Форма с таблицей накладных, здесь хранятся все данные о накладных. |
| Suplly.xaml.cs | Справочник поставок для редактирования информации администратором таблицы «Поставки», «Поставщик» и «Договор» |
| Employee.xaml.cs | Форма с таблицей сотрудников, здесь хранятся все данные сотрудниковпри регистрации. |
| MainWindow.xaml.cs | Основное меню для клиентов магазина с помощью которого, клиент переходит по вкладкам магазина. |
| Katalog.xaml.cs | Форма для выбора категории оружия для покупки. |
| Katalog2.xaml.cs | Форма для выбора модификаций для покупки. |
| Katalog1.xaml.cs | Форма для выбора патронов для покупки. |
| Kontaks.xaml.cs | Форма на котрой расположены контакты магазина. |
| Vakansii.xaml.cs | Форма со списком свободных вакансий в магазине. |
| PreyscourantWeaponTest.xaml.cs | Форма на которой клиент добавляет нужный ему товар в корзину и осуществляет печать чека. |
| AssauiltRifles.xaml.cs | Форма с изображением продаваемых штурмовых винтовок. |
| SMG.xaml.cs | Форма с изображением продаваемых пистолетов-пулемётов. |
| LMG.xaml.cs | Форма с изображением продаваемыхручных пулеметах. |
| SniperRifles.xaml.cs | Форма с изображением продаваемых снайперских винтовок. |
| Shotguns.xaml.cs | Форма с изображением продаваемых дробовиков. |
| Pistols.xsml.cs | Форма с изображением продаваемых пистолетов. |

## 1.4 Код структуры программы

Authorization\_Form.xaml.cs

sing WeaponStore.TableConnection;

using System;

using Microsoft.Win32;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace WeaponStore

{

/// <summary>

/// Логика взаимодействия для Authorization.xaml

/// </summary>

public partial class Authorization : Window

{

public Authorization()

{

InitializeComponent();

}

private void Button\_Click(object sender, RoutedEventArgs e)

{

Session.currentUser = null;

var users = new DBProcedures().getAuthorizationList();

foreach (var user in users)

{

if (user.Login == txtBoxLog.Text && user.Password == txtBoxPas.Password)

{

Session.currentUser = user;

AccountWindow clientPersonal = new AccountWindow();

clientPersonal.Show();

if (Session.mainWindow != null)

{

Session.mainWindow.Close();

}

Close();

return;

}

}

MessageBox.Show("Ошибка в логине или пароле!");

}

private void Button\_Click\_2(object sender, RoutedEventArgs e)

{

(new Registration(null)).ShowDialog();

}

}

}

Ammo.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

using System.Data;

using System.Data.SqlClient;

using WeaponStore.TableConnection;

namespace WeaponStore

{

/// <summary>

/// Логика взаимодействия для Ammo.xaml

/// </summary>

public partial class Ammo : Window

{

private TableConnection.ConnectionAmmo connectionAmmo;

List<TableConnection.ConnectionAmmo> Ammos;

public string extension = string.Empty;

private string QR = "";

public Ammo()

{

InitializeComponent();

}

DBProcedures procedures = new DBProcedures();

private void Window\_Activated(object sender, EventArgs e)

{

//Ammos= (new DBProcedures()).getAmmoList();

//dgAmmo.ItemsSource = Ammos;

//dgAmmo.Columns[0].Visibility = Visibility.Collapsed;

}

private void BtInsert\_Click(object sender, RoutedEventArgs e)

{

if (tbTypeAmmo.Text == string.Empty ||

tbAmmountAmmo.Text == string.Empty ||

tbCost.Text == string.Empty)

{

MessageBox.Show("Ошибка");

return;

}

new DBProcedures().spAmmo\_insert(new TableConnection.ConnectionAmmo(

-1,

tbTypeAmmo.Text,

Convert.ToInt32(tbAmmountAmmo.Text),

tbCost.Text

));

Ammos = (new DBProcedures()).getAmmoList();

dgAmmo.ItemsSource = Ammos;

dgAmmo.Columns[0].Visibility = Visibility.Collapsed;

}

private void BtUpdate\_Click(object sender, RoutedEventArgs e)

{

int selectedIndex = dgAmmo.SelectedIndex;

var ap = new DBProcedures().getAmmoList();

foreach (var it in ap)

{

if (it.ID\_Ammo == Ammos[selectedIndex].ID\_Ammo);

{

connectionAmmo = it;

break;

}

}

if (tbTypeAmmo.Text == string.Empty ||

tbAmmountAmmo.Text == string.Empty ||

tbCost.Text == string.Empty)

{

MessageBox.Show("Ошибка");

return;

}

new DBProcedures().spAmmo\_update(new TableConnection.ConnectionAmmo(

this.connectionAmmo.ID\_Ammo,

tbTypeAmmo.Text,

Convert.ToInt32(tbAmmountAmmo.Text),

tbCost.Text

));

Ammos = (new DBProcedures()).getAmmoList();

dgAmmo.ItemsSource = Ammos;

dgAmmo.Columns[0].Visibility = Visibility.Collapsed;

}

private void BtDelete\_Click(object sender, RoutedEventArgs e)

{

int selectedIndex = dgAmmo.SelectedIndex;

new DBProcedures().spAmmo\_delete(Ammos[selectedIndex].ID\_Ammo);

MessageBox.Show("Операция выполнена");

Ammos = (new DBProcedures()).getAmmoList();

dgAmmo.ItemsSource = Ammos;

dgAmmo.Columns[0].Visibility = Visibility.Collapsed;

}

private void BtSearch\_Click(object sender, RoutedEventArgs e)

{

foreach(DataRowView dataRow in (DataView)dgAmmo.ItemsSource)

{

if (dataRow.Row.ItemArray[1].ToString() == tbSearch.Text ||

dataRow.Row.ItemArray[2].ToString() == tbSearch.Text ||

dataRow.Row.ItemArray[3].ToString() == tbSearch.Text ||

dataRow.Row.ItemArray[4].ToString() == tbSearch.Text ||

dataRow.Row.ItemArray[7].ToString() == tbSearch.Text)

{

dgAmmo.SelectedItem = dataRow;

}

}

}

private void BtFilter\_Click(object sender, RoutedEventArgs e)

{

string newQR = QR + " where [Type\_Ammo] like '%" + tbSearch.Text + "%' or " +

"[Ammount\_Ammo] like '%" + tbSearch.Text + "%' or " +

"[Cost] like '%" + tbSearch.Text + "%'";

Ammos = (new DBProcedures()).getAmmoList();

dgAmmo.ItemsSource = Ammos;

}

private void BtBack\_Click(object sender, RoutedEventArgs e)

{

Spravochnik spravochnik = new Spravochnik();

spravochnik.Show();

Visibility = Visibility.Collapsed;

}

private void ChbdgAmmo\_Checked(object sender, RoutedEventArgs e)

{

dgAmmo.Visibility = Visibility.Visible;

}

private void ChbdgAmmo\_Unchecked(object sender, RoutedEventArgs e)

{

dgAmmo.Visibility = Visibility.Hidden;

}

private void Window\_Loaded(object sender, RoutedEventArgs e)

{

dgAmmo.Visibility = Visibility.Hidden;

Ammos = (new DBProcedures()).getAmmoList();

dgAmmo.ItemsSource = Ammos;

dgAmmo.Columns[0].Visibility = Visibility.Collapsed;

}

}

}}

AssaultRifles.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

using System.Data;

namespace WeaponStore

{

/// <summary>

/// Логика взаимодействия для AssaultRifles.xaml

/// </summary>

public partial class AssaultRifles : Window

{

public AssaultRifles()

{

InitializeComponent();

}

private void BtHemlok\_Click(object sender, RoutedEventArgs e)

{

PreyscourantWeaponTest preyscourantWeaponTest = new PreyscourantWeaponTest();

preyscourantWeaponTest.Show();

//preyscourantWeaponTest.BtSearch\_Click(null,null);

Visibility = Visibility.Collapsed;

}

private void BtFlatline\_Click(object sender, RoutedEventArgs e)

{

PreyscourantWeaponTest preyscourantWeaponTest = new PreyscourantWeaponTest();

preyscourantWeaponTest.Show();

Visibility = Visibility.Collapsed;

}

private void BtR\_301\_Click(object sender, RoutedEventArgs e)

{

PreyscourantWeaponTest preyscourantWeaponTest = new PreyscourantWeaponTest();

preyscourantWeaponTest.Show();

Visibility = Visibility.Collapsed;

}

private void Havok\_Click(object sender, RoutedEventArgs e)

{

PreyscourantWeaponTest preyscourantWeaponTest = new PreyscourantWeaponTest();

preyscourantWeaponTest.Show();

Visibility = Visibility.Collapsed;

}

}

}

BlockKlient.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace WeaponStore

{

/// <summary>

/// Логика взаимодействия для BlockKlient.xaml

/// </summary>

public partial class BlockKlient : Window

{

public BlockKlient()

{

InitializeComponent();

}

List<TableConnection.ConnectionKlientDetail> listKlientDetail;

private void updateData()

{

listKlientDetail = (new DBProcedures()).getKlientListDetail();

dataGridClients.ItemsSource = listKlientDetail;

dataGridClients.Columns[0].Visibility = Visibility.Collapsed;

}

private void Window\_Activated(object sender, EventArgs e)

{

//updateData();

}

private void ButtonClose\_Click(object sender, RoutedEventArgs e)

{

Spravochnik spravochnik = new Spravochnik();

spravochnik.Show();

Visibility = Visibility.Collapsed;

}

private void ButtonBlock\_Click(object sender, RoutedEventArgs e)

{

int selectedIndex = dataGridClients.SelectedIndex;

if (selectedIndex < 0)

{

MessageBox.Show("Не выбран клиент");

return;

}

var ap = new DBProcedures().getKlientList();

TableConnection.ConnectionKlient connectionKlient = null;

foreach (var it in ap)

{

if (it.ID\_Authorization == listKlientDetail[selectedIndex].ID\_Authorization)

{

connectionKlient = it;

break;

}

}

connectionKlient.License\_ID = 3;

new DBProcedures().spKlient\_update(connectionKlient);

MessageBox.Show("Операция выполнена");

updateData();

}

private void ButtonUnBlock\_Click(object sender, RoutedEventArgs e)

{

int selectedIndex = dataGridClients.SelectedIndex;

if (selectedIndex < 0)

{

MessageBox.Show("Не выбран клиент");

return;

}

var ap = new DBProcedures().getKlientList();

TableConnection.ConnectionKlient connectionKlient = null;

foreach (var it in ap)

{

if (it.ID\_Authorization == listKlientDetail[selectedIndex].ID\_Authorization)

{

connectionKlient = it;

break;

}

}

connectionKlient.License\_ID = 2;

new DBProcedures().spKlient\_update(connectionKlient);

MessageBox.Show("Операция выполнена");

updateData();

}

private void ChbdgBlockklient\_Checked(object sender, RoutedEventArgs e)

{

dataGridClients.Visibility = Visibility.Visible;

}

private void ChbdgBlockklient\_Unchecked(object sender, RoutedEventArgs e)

{

dataGridClients.Visibility = Visibility.Hidden;

}

private void Window\_Loaded(object sender, RoutedEventArgs e)

{

updateData();

dataGridClients.Visibility = Visibility.Hidden;

}

}

}

Employee.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

using System.Data;

using System.Data.SqlClient;

using WeaponStore.TableConnection;

using System.Collections.Generic;

using System.Collections.ObjectModel;

namespace WeaponStore

{

/// <summary>

/// Логика взаимодействия для Employee.xaml

/// </summary>

public partial class Employee : Window

{

private TableConnection.ConnectionEmployee connectionEmployee;

List<TableConnection.ConnectionEmployee> Employees;

private TableConnection.ConnectionEmployee1 connectionEmployee1;

List<TableConnection.ConnectionEmployee> Employees1;

private TableConnection.ConnectionPosition connectionPosition;

List<TableConnection.ConnectionPosition> Positions;

private List<TableConnection.ConnectionRole> roles;

public string extension = string.Empty;

private string QR = "";

public Employee()

{

InitializeComponent();

roles = new DBProcedures().getRoleList();

ObservableCollection<string> listPositions = new ObservableCollection<string>();

foreach (var it in roles)

{

listPositions.Add(it.Title\_Role);

}

this.lbPosition.ItemsSource = listPositions;

this.connectionEmployee1 = connectionEmployee1;

if (this.connectionEmployee != null)

{

tbFirstName.Text = this.connectionEmployee1.First\_Name\_Employee;

tbName.Text = this.connectionEmployee1.Name\_Employee;

tbMiddleName.Text = this.connectionEmployee1.Middle\_Name\_Employee;

tbJobExpirience.Text = this.connectionEmployee1.Job\_Experience;

tbEmploymentData.Text = this.connectionEmployee1.Employment\_Data;

lbPosition.IsEnabled = false;

txtBoxLogin.Visibility = Visibility.Hidden;

txtBoxPassword.Visibility = Visibility.Hidden;

txtBoxPassword2.Visibility = Visibility.Hidden;

labelPass.Visibility = Visibility.Hidden;

labelPass2.Visibility = Visibility.Hidden;

labelLogin.Visibility = Visibility.Hidden;

int idlistRoles = -1;

for (idlistRoles = 0; idlistRoles < roles.Count; idlistRoles++)

{

TableConnection.ConnectionAuthorization authorization = null;

foreach (var it in new DBProcedures().getAuthorizationList())

{

if (it.ID\_Authorization == this.connectionEmployee1.ID\_Authorization)

{

authorization = it;

break;

}

}

if (authorization != null)

{

if (roles[idlistRoles].ID\_Role == authorization.ID\_Role)

{

break;

}

}

}

lbPosition.SelectedIndex = idlistRoles;

}

else

{

tbName.Text = string.Empty;

tbMiddleName.Text = string.Empty;

tbJobExpirience.Text = string.Empty;

tbEmploymentData.Text = string.Empty;

}

}

DBProcedures procedures = new DBProcedures();

private void Window\_Activated(object sender, EventArgs e)

{

//Employees = (new DBProcedures()).getEmployeeList();

//dgEmployee.ItemsSource = Employees;

//Positions = (new DBProcedures()).getPositionList();

//lbPosition.ItemsSource = Positions;

}

private void dgFill(string qr)

{

ConnectionEmployee connection = new ConnectionEmployee();

ConnectionEmployee.qrEmployee = qr;

connection.Employee\_Fill();

dgEmployee.ItemsSource = connection.dtEmployee.DefaultView;

dgEmployee.Columns[0].Visibility = Visibility.Collapsed;

dgEmployee.Columns[6].Visibility = Visibility.Collapsed;

dgEmployee.Columns[7].Visibility = Visibility.Collapsed;

}

private void BtInsert\_Click(object sender, RoutedEventArgs e)

{

int idAuth(string Login)

{

var dbprocedures = new DBProcedures();

var auths = dbprocedures.getAuthorizationList();

foreach (var a in auths)

{

if (a.Login == txtBoxLogin.Text)

{

return a.ID\_Authorization;

}

}

return -1;

}

TableConnection.ConnectionAuthorization newAuthorization = new TableConnection.ConnectionAuthorization(-1, txtBoxLogin.Text, txtBoxPassword.Text, 4);

new DBProcedures().spAuthorization\_insert(newAuthorization);

int ID\_Position = Positions[lbPosition.SelectedIndex].ID\_Position;

int id\_Auth = idAuth(txtBoxLogin.Text);

TableConnection.ConnectionEmployee1 employee1 = new TableConnection.ConnectionEmployee1(

id\_Auth,

tbFirstName.Text,

tbName.Text,

tbMiddleName.Text,

tbJobExpirience.Text,

tbEmploymentData.Text,

ID\_Position

);

new DBProcedures().spEmployee\_insert1(employee1);

MessageBox.Show("Регистрация прошла успешно");

dgFill(QR);

//new DBProcedures().spEmployee\_insert(new TableConnection.ConnectionEmployee(

// -1,

// tbFirstName.Text,

// tbName.Text,

// tbMiddleName.Text,

// tbJobExpirience.Text,

// tbEmploymentData.Text,

// Convert.ToInt32(lbPosition.SelectedValue)

// ));

//Employees = (new DBProcedures()).getEmployeeList();

//dgEmployee.ItemsSource = Employees;

//procedures.spEmployee\_insert(idAuth,tbFirstName.Text,tbName.Text, tbMiddleName.Text, tbJobExpirience.Text,tbEmploymentData.Text ,Convert.ToInt32(lbPosition.SelectedValue));

}

private void BtUpdate\_Click(object sender, RoutedEventArgs e)

{

//new DBProcedures().spEmployee\_update(new TableConnection.ConnectionEmployee(

//this.connectionEmployee.ID\_Authorization,

// tbFirstName.Text,

// tbName.Text,

// tbMiddleName.Text,

// tbJobExpirience.Text,

// tbEmploymentData.Text,

// Convert.ToInt32(lbPosition.SelectedValue)

// ));

//Employees = (new DBProcedures()).getEmployeeList();

//dgEmployee.ItemsSource = Employees;

try

{

DataRowView ID = (DataRowView)dgEmployee.SelectedItems[0];

if (ID == null)

MessageBox.Show("", "");

else

procedures.spEmployee\_update(tbFirstName.Text, tbName.Text, tbMiddleName.Text, tbJobExpirience.Text, tbEmploymentData.Text, Convert.ToInt32(ID["ID\_Authorization"]), Convert.ToInt32(lbPosition.SelectedValue));

dgFill(QR);

}

catch

{

}

}

private void BtDelete\_Click(object sender, RoutedEventArgs e)

{

switch (MessageBox.Show("Удалить выбранную запись?",

"Удаление записи", MessageBoxButton.YesNo,

MessageBoxImage.Warning))

{

case MessageBoxResult.Yes:

DataRowView ID =

(DataRowView)dgEmployee.SelectedItems[0];

procedures.spEmployee\_delete(

Convert.ToInt32(ID["ID\_Authorization"]));

dgFill(QR);

break;

}

}

private void BtSearch\_Click(object sender, RoutedEventArgs e)

{

foreach (DataRowView dataRow in (DataView)dgEmployee.ItemsSource)

{

if (dataRow.Row.ItemArray[1].ToString() == tbSearch.Text ||

dataRow.Row.ItemArray[2].ToString() == tbSearch.Text ||

dataRow.Row.ItemArray[3].ToString() == tbSearch.Text ||

dataRow.Row.ItemArray[4].ToString() == tbSearch.Text ||

dataRow.Row.ItemArray[5].ToString() == tbSearch.Text ||

dataRow.Row.ItemArray[8].ToString() == tbSearch.Text ||

dataRow.Row.ItemArray[9].ToString() == tbSearch.Text)

{

dgEmployee.SelectedItem = dataRow;

}

}

}

private void BtFilter\_Click(object sender, RoutedEventArgs e)

{

switch (chbFilter.IsChecked)

{

case (true):

string newQR = QR + " where [First\_Name\_Employee] like '%" + tbSearch.Text + "%' or " +

"[Name\_Employee] like '%" + tbSearch.Text + "%' or " +

"[Middle\_Name\_Employee] like '%" + tbSearch.Text + "%' or " +

"[Job\_Experience] like '%" + tbSearch.Text + "%' or " +

"[Employment\_Data] like '%" + tbSearch.Text + "%' or " +

"[Position\_ID] like '%" + tbSearch.Text + "%'";

dgFill(newQR);

break;

case (false):

dgFill(QR);

break;

}

}

private void BtInsertPosition\_Click(object sender, RoutedEventArgs e)

{

new DBProcedures().spPosition\_insert(new TableConnection.ConnectionPosition(

-1,

tbName\_Position.Text,

tbSalary.Text

));

Positions = (new DBProcedures()).getPositionList();

lbPosition.ItemsSource = Positions;

}

private void BtUpdatePosition\_Click(object sender, RoutedEventArgs e)

{

new DBProcedures().spPosition\_update(new TableConnection.ConnectionPosition(

this.connectionPosition.ID\_Position,

tbName\_Position.Text,

tbSalary.Text

));

Positions = (new DBProcedures()).getPositionList();

lbPosition.ItemsSource = Positions;

}

private void BtDeletePosition\_Click(object sender, RoutedEventArgs e)

{

switch (MessageBox.Show("Удалить выбранную запись?",

"Удаление записи", MessageBoxButton.YesNo,

MessageBoxImage.Warning))

{

case MessageBoxResult.Yes:

procedures.spPosition\_delete(

Convert.ToInt32(lbPosition.

SelectedValue.ToString()));

Positions = (new DBProcedures()).getPositionList();

lbPosition.ItemsSource = Positions;

break;

}

}

private void Window\_Loaded(object sender, RoutedEventArgs e)

{

dgEmployee.Visibility = Visibility.Hidden;

QR = ConnectionEmployee.qrEmployee;

dgFill(QR);

Positions = (new DBProcedures()).getPositionList();

lbPosition.ItemsSource = Positions;

lbPosition.SelectedValuePath = "ID\_Position";

lbPosition.DisplayMemberPath = "Name\_Position";

}

private void BtBack\_Click(object sender, RoutedEventArgs e)

{

Spravochnik spravochnik = new Spravochnik();

spravochnik.Show();

Visibility = Visibility.Collapsed;

}

private void ChbdgNakladnaya\_Checked(object sender, RoutedEventArgs e)

{

dgEmployee.Visibility = Visibility.Visible;

}

private void ChbdgNakladnaya\_Unchecked(object sender, RoutedEventArgs e)

{

dgEmployee.Visibility = Visibility.Hidden;

}

}

}

Katalog.xaml

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace WeaponStore

{

/// <summary>

/// Логика взаимодействия для Katalog.xaml

/// </summary>

public partial class Katalog : Window

{

public Katalog()

{

InitializeComponent();

}

private void BtAssault\_Rifles\_Click(object sender, RoutedEventArgs e)

{

AssaultRifles assaultRifles = new AssaultRifles();

assaultRifles.Show();

Visibility = Visibility.Collapsed;

}

private void BtSubMuchineGuns\_Click(object sender, RoutedEventArgs e)

{

SMG sMG = new SMG();

sMG.Show();

Visibility = Visibility.Collapsed;

}

private void MuchineGuns\_Click(object sender, RoutedEventArgs e)

{

LMG lMG = new LMG();

lMG.Show();

Visibility = Visibility.Collapsed;

}

private void SniperRifles\_Click(object sender, RoutedEventArgs e)

{

SniperRifles sniperRifles = new SniperRifles();

sniperRifles.Show();

Visibility = Visibility.Collapsed;

}

private void Shotguns\_Click(object sender, RoutedEventArgs e)

{

Shotguns shotguns = new Shotguns();

shotguns.Show();

Visibility = Visibility.Collapsed;

}

private void Pistols\_Click(object sender, RoutedEventArgs e)

{

Pistols pistols = new Pistols();

pistols.Show();

Visibility = Visibility.Collapsed;

}

private void Weapon\_Click(object sender, RoutedEventArgs e)

{

MessageBox.Show("Вы находитесь на этой вкладке");

return;

}

private void Ammo\_Click(object sender, RoutedEventArgs e)

{

Katalog1 katalog1 = new Katalog1();

katalog1.Show();

Visibility = Visibility.Collapsed;

//Test test = new Test();

//test.Show();

//Visibility = Visibility.Collapsed;

}

private void Modifications\_Click(object sender, RoutedEventArgs e)

{

Katalog2 katalog2 = new Katalog2();

katalog2.Show();

Visibility = Visibility.Collapsed;

}

private void BtBack\_Click(object sender, RoutedEventArgs e)

{

MainWindow mainWindow = new MainWindow();

mainWindow.Show();

Visibility = Visibility.Collapsed;

}

}

}

Katalog1.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace WeaponStore

{

/// <summary>

/// Логика взаимодействия для Katalog1.xaml

/// </summary>

public partial class Katalog1 : Window

{

public Katalog1()

{

InitializeComponent();

}

private void Weapon\_Click(object sender, RoutedEventArgs e)

{

Katalog katalog = new Katalog();

katalog.Show();

Visibility = Visibility.Collapsed;

}

private void Ammo\_Click(object sender, RoutedEventArgs e)

{

MessageBox.Show("Вы находитесь на этой вкладке");

return;

}

private void Modifications\_Click(object sender, RoutedEventArgs e)

{

Katalog2 katalog2 = new Katalog2();

katalog2.Show();

Visibility = Visibility.Collapsed;

}

private void BtShotgun\_ammo\_Click(object sender, RoutedEventArgs e)

{

MessageBox.Show("Ознакомьтесь с информацией для клиентов!В ближайшее время черный рынок возобновит свою работу.");

}

private void BtHeavy\_ammo\_Click(object sender, RoutedEventArgs e)

{

MessageBox.Show("Ознакомьтесь с информацией для клиентов!В ближайшее время черный рынок возобновит свою работу.");

}

private void BtLight\_ammo\_Click(object sender, RoutedEventArgs e)

{

MessageBox.Show("Ознакомьтесь с информацией для клиентов!В ближайшее время черный рынок возобновит свою работу.");

}

private void BtSniper\_ammo\_Click(object sender, RoutedEventArgs e)

{

MessageBox.Show("Ознакомьтесь с информацией для клиентов!В ближайшее время черный рынок возобновит свою работу.");

}

private void BtBack\_Click(object sender, RoutedEventArgs e)

{

MainWindow mainWindow = new MainWindow();

mainWindow.Show();

Visibility = Visibility.Collapsed;

}

}

}

Katalog2.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace WeaponStore

{

/// <summary>

/// Логика взаимодействия для Katalog2.xaml

/// </summary>

public partial class Katalog2 : Window

{

public Katalog2()

{

InitializeComponent();

}

private void Weapon\_Click(object sender, RoutedEventArgs e)

{

Katalog katalog = new Katalog();

katalog.Show();

Visibility = Visibility.Collapsed;

}

private void Ammo\_Click(object sender, RoutedEventArgs e)

{

Katalog1 katalog1 = new Katalog1();

katalog1.Show();

Visibility = Visibility.Collapsed;

}

private void Modifications\_Click(object sender, RoutedEventArgs e)

{

MessageBox.Show("Вы находитесь на этой вкладке");

return;

}

private void Buy\_Click(object sender, RoutedEventArgs e)

{

//PreyscourantModifications preyscourantModifications = new PreyscourantModifications();

//preyscourantModifications.Show();

//Visibility = Visibility.Collapsed;

MessageBox.Show("Ознакомьтесь с информацией для клиентов!В ближайшее время черный рынок возобновит свою работу.");

}

private void BtBack\_Click(object sender, RoutedEventArgs e)

{

MainWindow mainWindow = new MainWindow();

mainWindow.Show();

Visibility = Visibility.Collapsed;

}

}

KlientInfo.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace WeaponStore

{

/// <summary>

/// Логика взаимодействия для KlientInfo.xaml

/// </summary>

public partial class KlientInfo : Window

{

public KlientInfo()

{

InitializeComponent();

}

private void BtBack\_Click(object sender, RoutedEventArgs e)

{

Close();

}

}

}

Klients.xaml.cs

sing System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using Xceed.Words.NET;

using Xceed.Document.NET;

using Word = Microsoft.Office.Interop.Word;

using Excel = Microsoft.Office.Interop.Excel;

using System.Threading;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

using System.Diagnostics;

using System.IO;

namespace WeaponStore

{

/// <summary>

/// Логика взаимодействия для Klients.xaml

/// </summary>

public partial class Klients : Window

{

List<TableConnection.ConnectionKlientDetail> listKlient;

public string extension = string.Empty;

private void updateData()

{

listKlient = (new DBProcedures()).getKlientListDetail();

dataGridClients.ItemsSource = listKlient;

dataGridClients.Columns[0].Visibility = Visibility.Collapsed;

}

public Klients()

{

InitializeComponent();

}

DBProcedures procedures = new DBProcedures();

private void Window\_Activated(object sender, EventArgs e)

{

//updateData();

}

private void ButtonDetail\_Click(object sender, RoutedEventArgs e)

{

int selectedIndex = dataGridClients.SelectedIndex;

if (selectedIndex < 0)

{

MessageBox.Show("Не выбран клиент");

return;

}

var ap = new DBProcedures().getKlientList();

TableConnection.ConnectionKlient connectionKlient = null;

foreach (var it in ap)

{

if (it.ID\_Authorization == listKlient[selectedIndex].ID\_Authorization)

{

connectionKlient = it;

break;

}

}

var ce = new Registration(connectionKlient, true);

ce.ShowDialog();

}

private void ButtonClose\_Click(object sender, RoutedEventArgs e)

{

var roleList = new DBProcedures().getRoleList(Session.currentUser.ID\_Role);

InitializeComponent();

if (roleList.Count == 1)

{

var role = roleList[0];

this.Title = role.Title\_Role;

if (role.Admin == 1)

{

Spravochnik spravochnik = new Spravochnik();

spravochnik.Show();

Visibility = Visibility.Collapsed;

}

else

{

Close();

}

}

}

private void createExportDoc()

{

try

{

DBProcedures con = new DBProcedures();

var connectionKlientDetail = con.getKlientListDetail();

if (extension == string.Empty)

{

MessageBox.Show("Не выбран тип экспортруемого файла");

return;

}

switch (extension)

{

case (".docx"):

string pathDocumentDOCX = Session.baseDir + "Список клиентов" + extension;

DocX document = DocX.Create(pathDocumentDOCX);

Xceed.Document.NET.Paragraph paragraph = document.InsertParagraph();

paragraph.

AppendLine("Документ '" + "Отчет список клиентов" + "' создан " + DateTime.Now.ToShortDateString()).

Font("Time New Roman").

FontSize(16).Bold().Alignment = Alignment.left;

paragraph.AppendLine();

Xceed.Document.NET.Table doctable = document.AddTable(connectionKlientDetail.Count + 1, 2);

doctable.Design = TableDesign.TableGrid;

doctable.TableCaption = "Список клиентов";

doctable.Rows[0].Cells[0].Paragraphs[0].Append("Список клиентов").Font("Times New Roman").FontSize(14);

for (int i = 0; i < connectionKlientDetail.Count; i++)

{

doctable.Rows[i + 1].Cells[0].Paragraphs[0].Append(connectionKlientDetail[i].KlientInfo).Font("Times New Roman").FontSize(14);

}

document.InsertParagraph().InsertTableAfterSelf(doctable);

document.Save();

MessageBox.Show("Отчет успешно сформирован!");

Process.Start(pathDocumentDOCX);

break;

case (".xlsx"):

Excel.Application excel;

Excel.Workbook worKbooK;

Excel.Worksheet worKsheeT;

Excel.Range celLrangE;

string pathDocumentXLSX = Session.baseDir + "Список клиентов" + extension;

try

{

excel = new Excel.Application();

excel.Visible = false;

excel.DisplayAlerts = false;

worKbooK = excel.Workbooks.Add(Type.Missing);

worKsheeT = (Microsoft.Office.Interop.Excel.Worksheet)worKbooK.ActiveSheet;

worKsheeT.Name = "Список клиентов";

worKsheeT.Range[worKsheeT.Cells[1, 1], worKsheeT.Cells[1, 8]].Merge();

worKsheeT.Cells[1, 1] = "Список клиентов";

worKsheeT.Cells.Font.Size = 15;

for (int i = 0; i < connectionKlientDetail.Count; i++)

{

worKsheeT.Cells[i + 3, 1] = connectionKlientDetail[i].KlientInfo;

}

worKbooK.SaveAs(pathDocumentXLSX); ;

worKbooK.Close();

excel.Quit();

MessageBox.Show("Отчет успешно сформирован!");

Process.Start(pathDocumentXLSX);

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

finally

{

worKsheeT = null;

celLrangE = null;

worKbooK = null;

}

break;

case (".pdf"):

string pathDocumentPDF = Session.baseDir + "Список клиентов" + extension;

if (File.Exists(Session.baseDir + "Список клиентов.docx"))

{

Word.Application appWord = new Word.Application();

var wordDocument = appWord.Documents.Open(Session.baseDir + "Список клиентов.docx");

wordDocument.ExportAsFixedFormat(pathDocumentPDF, Word.WdExportFormat.wdExportFormatPDF);

MessageBox.Show("Отчет успешно сформирован!");

wordDocument.Close();

Process.Start(pathDocumentPDF);

}

else

MessageBox.Show("Сначала сформируйте отчет .docx");

break;

}

}

catch (Exception)

{

MessageBox.Show("Отсутсвие Ms Office на компьютере. Пожалуйста скачайте его.");

Process.Start("https://www.microsoft.com/ru-ru/microsoft-365/compare-all-microsoft-365-products?tab=1&rtc=1");

}

}

private void ButtonExport\_Click(object sender, RoutedEventArgs e)

{

Thread t = new Thread(new ThreadStart(createExportDoc));

t.Start();

}

private void ComboBoxExport\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

ComboBoxItem typeItem = (ComboBoxItem)comboBoxExport.SelectedItem;

extension = typeItem.Content.ToString();

}

private void ChdgKlients\_Checked(object sender, RoutedEventArgs e)

{

dataGridClients.Visibility = Visibility.Visible;

}

private void ChdgKlients\_Unchecked(object sender, RoutedEventArgs e)

{

dataGridClients.Visibility = Visibility.Hidden;

}

private void Window\_Loaded(object sender, RoutedEventArgs e)

{

dataGridClients.Visibility = Visibility.Hidden;

updateData();

}

}

}

Kontacts.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace WeaponStore

{

/// <summary>

/// Логика взаимодействия для Kontacts.xaml

/// </summary>

public partial class Kontacts : Window

{

public Kontacts()

{

InitializeComponent();

}

private void Button\_Click(object sender, RoutedEventArgs e)

{

MainWindow mainWindow = new MainWindow();

mainWindow.Show();

Visibility = Visibility.Collapsed;

}

}

}

LMG.xaml.cs

using System;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace WeaponStore

{

/// <summary>

/// Логика взаимодействия для LMG.xaml

/// </summary>

public partial class LMG : Window

{

public LMG()

{

InitializeComponent();

}

private void BtDevotion\_Click(object sender, RoutedEventArgs e)

{

PreyscourantWeaponTest preyscourantWeaponTest = new PreyscourantWeaponTest();

preyscourantWeaponTest.Show();

Visibility = Visibility.Collapsed;

}

private void BtSpitfire\_Click(object sender, RoutedEventArgs e)

{

PreyscourantWeaponTest preyscourantWeaponTest = new PreyscourantWeaponTest();

preyscourantWeaponTest.Show();

Visibility = Visibility.Collapsed;

}

private void BtL\_Star\_Click(object sender, RoutedEventArgs e)

{

PreyscourantWeaponTest preyscourantWeaponTest = new PreyscourantWeaponTest();

preyscourantWeaponTest.Show();

Visibility = Visibility.Collapsed;

}

}

}

MainWindow.xaml

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace WeaponStore

{

/// <summary>

/// Логика взаимодействия для MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

}

private void Window\_Activated(object sender, EventArgs e)

{

InitializeComponent();

Session.mainWindow = this;

}

private void Button\_Click(object sender, RoutedEventArgs e)

{

Katalog katalog = new Katalog();

katalog.Show();

Visibility = Visibility.Collapsed;

}

private void Button\_Click\_1(object sender, RoutedEventArgs e)

{

Kontacts kontacts = new Kontacts();

kontacts.Show();

Visibility = Visibility.Collapsed;

}

private void Button\_Click\_2(object sender, RoutedEventArgs e)

{

//Session.currentUser = null;

//var roleList = new DBProcedures().getRoleList(Session.currentUser.ID\_Role);

//if (roleList.Count == 1)

//{

// var role = roleList[0];

// this.Title = role.Title\_Role;

// switch (role.Title\_Role == "Администратор")

// {

// case (true):

//Spravochnik spravochnik = new Spravochnik();

//spravochnik.Show();

//Visibility = Visibility.Collapsed;

// break;

// case (false):

// MessageBox.Show("Вы не администратор");

// break;

// }

//}

MessageBox.Show("Вы не администратор");

}

private void Button\_Click\_3(object sender, RoutedEventArgs e)

{

Vakansii vakansii = new Vakansii();

vakansii.Show();

Visibility = Visibility.Collapsed;

}

}

}

Modifications.xaml.cs

using System;

using System.Collections.Generic;

using System.Collections.ObjectModel;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

using System.Data;

namespace WeaponStore

{

/// <summary>

/// Логика взаимодействия для Modifications.xaml

/// </summary>

public partial class Modifications : Window

{

private TableConnection.ConnectionModifications connectionModifications;

List<TableConnection.ConnectionModifications> Modification;

public string extension = string.Empty;

private string QR = "";

public Modifications()

{

InitializeComponent();

}

//public Modifications(TableConnection.ConnectionModifications connectionModifications)

//{

// InitializeComponent();

//}

DBProcedures procedures = new DBProcedures();

private void BtInsert\_Click(object sender, RoutedEventArgs e)

{

if (tbNameModification.Text == string.Empty ||

tbAmmountModifications.Text == string.Empty ||

tbCost.Text == string.Empty)

{

MessageBox.Show("Ошибка");

return;

}

new DBProcedures().spModifications\_insert(new TableConnection.ConnectionModifications(

-1,

tbNameModification.Text,

Convert.ToInt32(tbAmmountModifications.Text),

tbCost.Text

));

Modification = (new DBProcedures()).getModificationsList();

dgModifications.ItemsSource = Modification;

dgModifications.Columns[0].Visibility = Visibility.Collapsed;

}

private void Window\_Activated(object sender, EventArgs e)

{

//dgModifications.Visibility = Visibility.Hidden;

//Modification = (new DBProcedures()).getModificationsList();

//dgModifications.ItemsSource = Modification;

//dgModifications.Columns[0].Visibility = Visibility.Collapsed;

}

private void BtUpdate\_Click(object sender, RoutedEventArgs e)

{

int selectedIndex = dgModifications.SelectedIndex;

var ap = new DBProcedures().getModificationsList();

foreach (var it in ap)

{

if (it.ID\_Modification == Modification[selectedIndex].ID\_Modification) ;

{

connectionModifications = it;

break;

}

}

if (tbNameModification.Text == string.Empty ||

tbAmmountModifications.Text == string.Empty ||

tbCost.Text == string.Empty)

{

MessageBox.Show("Ошибка");

return;

}

new DBProcedures().spModifications\_update(new TableConnection.ConnectionModifications(

this.connectionModifications.ID\_Modification,

tbNameModification.Text,

Convert.ToInt32(tbAmmountModifications.Text),

tbCost.Text

));

Modification = (new DBProcedures()).getModificationsList();

dgModifications.ItemsSource = Modification;

dgModifications.Columns[0].Visibility = Visibility.Collapsed;

}

private void BtDelete\_Click(object sender, RoutedEventArgs e)

{

int selectedIndex = dgModifications.SelectedIndex;

new DBProcedures().spModifications\_delete(Modification[selectedIndex].ID\_Modification);

MessageBox.Show("Операция выполнена");

Modification = (new DBProcedures()).getModificationsList();

dgModifications.ItemsSource = Modification;

dgModifications.Columns[0].Visibility = Visibility.Collapsed;

}

private void BtSearch\_Click(object sender, RoutedEventArgs e)

{

foreach (DataRowView dataRow in (DataView)dgModifications.ItemsSource)

{

if (dataRow.Row.ItemArray[1].ToString() == tbSearch.Text ||

dataRow.Row.ItemArray[2].ToString() == tbSearch.Text ||

dataRow.Row.ItemArray[3].ToString() == tbSearch.Text ||

dataRow.Row.ItemArray[4].ToString() == tbSearch.Text ||

dataRow.Row.ItemArray[7].ToString() == tbSearch.Text)

{

dgModifications.SelectedItem = dataRow;

}

}

}

private void BtFilter\_Click(object sender, RoutedEventArgs e)

{

string newQR = QR + " where [Name\_Modification] like '%" + tbSearch.Text + "%' or " +

"[Ammount\_Modifications] like '%" + tbSearch.Text + "%' or " +

"[Cost] like '%" + tbSearch.Text + "%'";

Modification = (new DBProcedures()).getModificationsList();

dgModifications.ItemsSource = Modification;

}

private void BtBack\_Click(object sender, RoutedEventArgs e)

{

Spravochnik spravochnik = new Spravochnik();

spravochnik.Show();

Visibility = Visibility.Collapsed;

}

private void ChbdgModifications\_Checked(object sender, RoutedEventArgs e)

{

dgModifications.Visibility = Visibility.Visible;

}

private void ChbdgModifications\_Unchecked(object sender, RoutedEventArgs e)

{

dgModifications.Visibility = Visibility.Hidden;

}

private void Window\_Loaded(object sender, RoutedEventArgs e)

{

dgModifications.Visibility = Visibility.Hidden;

Modification = (new DBProcedures()).getModificationsList();

dgModifications.ItemsSource = Modification;

dgModifications.Columns[0].Visibility = Visibility.Collapsed;

}

}

}

Nakladnaya.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

using System.Data;

using System.Data.SqlClient;

using WeaponStore.TableConnection;

namespace WeaponStore

{

/// <summary>

/// Логика взаимодействия для Nakladnaya.xaml

/// </summary>

public partial class Nakladnaya : Window

{

//private TableConnection.ConnectionNakladnaya connectionNakladnaya;

//List<TableConnection.ConnectionNakladnaya> Nakladnayas;

//private TableConnection.ConnectionSupply connectionSupply;

//List<TableConnection.ConnectionSupply> Supplies;

public string extension = string.Empty;

private string QR = "";

public Nakladnaya()

{

InitializeComponent();

}

DBProcedures procedures = new DBProcedures();

private void Window\_Activated(object sender, EventArgs e)

{

//Nakladnayas = (new DBProcedures()).getNakladnayaList();

//dgNakladnaya.ItemsSource = Nakladnayas;

//Supplies = (new DBProcedures()).getSupplyList();

//lbSupply.ItemsSource = Supplies;

}

private void dgFill(string qr)

{

ConnectionNakladnaya connection = new ConnectionNakladnaya();

ConnectionNakladnaya.qrNakladnaya = qr;

connection.Nakladnaya\_Fill();

dgNakladnaya.ItemsSource = connection.dtNakladnaya.DefaultView;

dgNakladnaya.Columns[0].Visibility = Visibility.Collapsed;

dgNakladnaya.Columns[2].Visibility = Visibility.Collapsed;

dgNakladnaya.Columns[3].Visibility = Visibility.Collapsed;

dgNakladnaya.Columns[8].Visibility = Visibility.Collapsed;

dgNakladnaya.Columns[9].Visibility = Visibility.Collapsed;

dgNakladnaya.Columns[11].Visibility = Visibility.Collapsed;

dgNakladnaya.Columns[12].Visibility = Visibility.Collapsed;

}

private void lbFill()

{

ConnectionSupply connection = new ConnectionSupply();

connection.Supply\_Fill();

lbSupply.ItemsSource

= connection.dtSupply.DefaultView;

lbSupply.SelectedValuePath = "ID\_Supply";

lbSupply.DisplayMemberPath = "Дата\_поставки";

}

private void BtInsert\_Click(object sender, RoutedEventArgs e)

{

//new DBProcedures().spNakladnaya\_insert(new TableConnection.ConnectionNakladnaya(

// -1,

// tbNumberNakladnaya.Text,

// Convert.ToInt32(lbSupply.SelectedValue)

// ));

//Nakladnayas = (new DBProcedures()).getNakladnayaList();

//dgNakladnaya.ItemsSource = Nakladnayas;

procedures.spNakladnaya\_insert(tbNumberNakladnaya.Text, Convert.ToInt32(lbSupply.SelectedValue));

dgFill(QR);

}

private void BtUpdate\_Click(object sender, RoutedEventArgs e)

{

//new DBProcedures().spNakladnaya\_update(new TableConnection.ConnectionNakladnaya(

//this.connectionNakladnaya.ID\_Nakladnaya,

//tbNumberNakladnaya.Text,

// Convert.ToInt32(lbSupply.SelectedValue)

// ));

//Nakladnayas = (new DBProcedures()).getNakladnayaList();

//dgNakladnaya.ItemsSource = Nakladnayas;

try

{

DataRowView ID = (DataRowView)dgNakladnaya.SelectedItems[0];

if (ID == null)

MessageBox.Show("", "");

else

procedures.spNakladnaya\_update(tbNumberNakladnaya.Text, Convert.ToInt32(ID["ID\_Nakladnaya"]), Convert.ToInt32(lbSupply.SelectedValue));

dgFill(QR);

}

catch

{

}

}

private void BtDelete\_Click(object sender, RoutedEventArgs e)

{

//switch (MessageBox.Show("Удалить выбранную запись?",

// "Удаление записи", MessageBoxButton.YesNo,

// MessageBoxImage.Warning))

//{

// case MessageBoxResult.Yes:

// procedures.spNakladnaya\_delete(

// Convert.ToInt32(dgNakladnaya.

// SelectedValue.ToString()));

// Nakladnayas = (new DBProcedures()).getNakladnayaList();

// dgNakladnaya.ItemsSource = Nakladnayas;

// break;

//}

switch (MessageBox.Show("Удалить выбранную запись?",

"Удаление записи", MessageBoxButton.YesNo,

MessageBoxImage.Warning))

{

case MessageBoxResult.Yes:

DataRowView ID =

(DataRowView)dgNakladnaya.SelectedItems[0];

procedures.spNakladnaya\_delete(

Convert.ToInt32(ID["ID\_Nakladnaya"]));

dgFill(QR);

break;

}

}

private void BtSearch\_Click(object sender, RoutedEventArgs e)

{

foreach (DataRowView dataRow in (DataView)dgNakladnaya.ItemsSource)

{

if (dataRow.Row.ItemArray[1].ToString() == tbSearch.Text ||

dataRow.Row.ItemArray[4].ToString() == tbSearch.Text ||

dataRow.Row.ItemArray[5].ToString() == tbSearch.Text ||

dataRow.Row.ItemArray[6].ToString() == tbSearch.Text ||

dataRow.Row.ItemArray[7].ToString() == tbSearch.Text ||

dataRow.Row.ItemArray[10].ToString() == tbSearch.Text ||

dataRow.Row.ItemArray[13].ToString() == tbSearch.Text)

{

dgNakladnaya.SelectedItem = dataRow;

}

}

}

private void BtFilter\_Click(object sender, RoutedEventArgs e)

{

switch (chbFilter.IsChecked)

{

case (true):

string newQR = QR + " where [Number\_Nakladnaya] like '%" + tbSearch.Text + "%' or " +

"[Supply\_ID] like '%" + tbSearch.Text + "%'";

dgFill(newQR);

break;

case (false):

dgFill(QR);

break;

}

}

private void Window\_Loaded(object sender, RoutedEventArgs e)

{

dgNakladnaya.Visibility = Visibility.Hidden;

QR = ConnectionNakladnaya.qrNakladnaya;

dgFill(QR);

lbFill();

}

private void BtBack\_Click(object sender, RoutedEventArgs e)

{

Spravochnik spravochnik = new Spravochnik();

spravochnik.Show();

Visibility = Visibility.Collapsed;

}

private void ChbdgNakladnaya\_Checked(object sender, RoutedEventArgs e)

{

dgNakladnaya.Visibility = Visibility.Visible;

}

private void ChbdgNakladnaya\_Unchecked(object sender, RoutedEventArgs e)

{

dgNakladnaya.Visibility = Visibility.Hidden;

}

}

}

Pistols.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace WeaponStore

{

/// <summary>

/// Логика взаимодействия для Pistols.xaml

/// </summary>

public partial class Pistols : Window

{

public Pistols()

{

InitializeComponent();

}

private void BtWingman\_Click(object sender, RoutedEventArgs e)

{

PreyscourantWeaponTest preyscourantWeaponTest = new PreyscourantWeaponTest();

preyscourantWeaponTest.Show();

Visibility = Visibility.Collapsed;

}

private void BtP\_2020\_Click(object sender, RoutedEventArgs e)

{

PreyscourantWeaponTest preyscourantWeaponTest = new PreyscourantWeaponTest();

preyscourantWeaponTest.Show();

Visibility = Visibility.Collapsed;

}

private void BtRE\_45\_Click(object sender, RoutedEventArgs e)

{

PreyscourantWeaponTest preyscourantWeaponTest = new PreyscourantWeaponTest();

preyscourantWeaponTest.Show();

Visibility = Visibility.Collapsed;

}

}

}

PreyscourantWeaponTest.xaml.cs

using System;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

using System.Data;

using System.Data.SqlClient;

using WeaponStore.TableConnection;

namespace WeaponStore

{

/// <summary>

/// Логика взаимодействия для PreyscourantWeaponTest.xaml

/// </summary>

public partial class PreyscourantWeaponTest : Window

{

List<TableConnection.ConnectionWeaponDetail> ListWeapons;

private TableConnection.ConnectionWeaponDetail connectionWeapon;

public string extension = string.Empty;

private string QR = "";

int purshace\_item = 0;

int[] count = new int[9999]; int step = 0;

int[] pushcar = new int[9999];

string korz;

public PreyscourantWeaponTest()

{

InitializeComponent();

}

private void Button\_Click(object sender, RoutedEventArgs e)

{

count[step] = 1;

step++;

MessageBox.Show("Товар добавлен в корзину!");

purshace\_item++;

Car.Header = "Корзина(" + purshace\_item.ToString() + ")";

pushcar[1]++;

count[step++] = 0;

//TableConnection.ConnectionWeapon connectionWeapon = dgWeapon.SelectedItem as TableConnection.ConnectionWeapon;

//string ID\_Weapon = Convert.ToString(connectionWeapon.ID\_Weapon);

//string Name\_Weapon = connectionWeapon.Name\_Weapon;

//string Accuracy = connectionWeapon.Accuracy;

//string Fire\_Rate = connectionWeapon.Fire\_Rate;

//string Shells\_In\_Store = connectionWeapon.Shells\_In\_Store;

//string Ammount\_Weapon = Convert.ToString(connectionWeapon.Ammount\_Weapon);

//string Cost = connectionWeapon.Cost;

//string Type\_Weapon\_ID = Convert.ToString(connectionWeapon.Type\_Weapon\_ID);

tbKorz.Text =tbKorz.Text.ToString() +" "+ tbPrey.Text.ToString() + " " + tbAccuracy.Text.ToString() + " " + tbFire\_Rate.Text.ToString() + " " + tbShells\_in\_store.Text.ToString() + " " + tbAmmount\_Weapon.Text.ToString() + " " + tbCost.Text.ToString()+Environment.NewLine;

//int result;

//int summ;

//result = Convert.ToInt32(tbCost.Text) \* Convert.ToInt32(tbAmmount\_Weapon.Text);

//summ = Convert.ToInt32(tbSumm.Text) + result;

//tbSumm.Text = Convert.ToString(summ);

Double db1, db2, db3,result,result2;

db1 = Double.Parse(tbCost.Text);

db2 = Double.Parse(tbAmmount\_Weapon.Text);

db3 = Double.Parse(tbSumm.Text);

result = db1 \* db2;

result2 = db3 + result;

tbSumm.Text =Convert.ToString(result2);

}

private void Window\_Activated(object sender, EventArgs e)

{

//ListWeapons = (new DBProcedures()).getWeaponListDetail();

//dgWeapon.ItemsSource = ListWeapons;

}

private void Button\_Click\_1(object sender, RoutedEventArgs e)

{

Katalog katalog = new Katalog();

katalog.Show();

Visibility = Visibility.Collapsed;

}

private void Window\_Loaded(object sender, RoutedEventArgs e)

{

dgWeapon.Visibility = Visibility.Hidden;

ListWeapons = (new DBProcedures()).getWeaponListDetail();

dgWeapon.ItemsSource = ListWeapons;

dgWeapon.Columns[0].Visibility = Visibility.Collapsed;

//QR = ConnectionWeapon1.qrWeapon;

//dgFill(QR);

}

private void Button\_Click\_2(object sender, RoutedEventArgs e)

{

//if ((Convert.ToInt32(tbAmmount\_Weapon.Text)) > connectionWeapon.Ammount\_Weapon)

// MessageBox.Show("На складе больше нет столько товара");

//else

int selectedIndex = dgWeapon.SelectedIndex;

var ap = new DBProcedures().getWeaponListDetail();

foreach (var it in ap)

{

if (it.ID\_Weapon == ListWeapons[selectedIndex].ID\_Weapon);

{

connectionWeapon = it;

break;

}

}

switch ((Convert.ToInt32(tbAmmount\_Weapon.Text)) >= connectionWeapon.Ammount\_Weapon)

{

case (true):

MessageBox.Show("Столько товара нет на складе");

break;

case (false):

lbAmmount.Visibility = Visibility.Visible;

tbAmmount\_Weapon.Visibility = Visibility.Visible;

int a = 1;

int b = Convert.ToInt32(tbAmmount\_Weapon.Text);

int result = a + b;

tbAmmount\_Weapon.Text = Convert.ToString(result);

break;

}

}

private void Button\_Click\_3(object sender, RoutedEventArgs e)

{

switch ((Convert.ToInt32(tbAmmount\_Weapon.Text)) <= 1)

{

case (true):

System.Windows.Forms.MessageBox.Show("Значение не может быть меньше 1");

break;

case (false):

lbAmmount.Visibility = Visibility.Visible;

tbAmmount\_Weapon.Visibility = Visibility.Visible;

int a = 1;

int p = Convert.ToInt32(tbAmmount\_Weapon.Text);

tbAmmount\_Weapon.Text = Convert.ToString(p - a);

break;

}

}

private void BtPrint\_Click(object sender, RoutedEventArgs e)

{

System.Windows.Controls.PrintDialog p = new System.Windows.Controls.PrintDialog();

if (p.ShowDialog() == true)

{

p.PrintVisual(grid1, "Печать");

}

}

public void BtSearch\_Click(object sender, RoutedEventArgs e)

{

//string newQR = QR + "[Name\_Type\_Weapon] like '%" +"Shotgun" + "%'"; ;

//dgFill(newQR);

}

private void dgFill(string qr)

{

ConnectionWeapon1 connection = new ConnectionWeapon1();

ConnectionWeapon1.qrWeapon = qr;

connection.Weapon\_Fill();

dgWeapon.ItemsSource = connection.dtWeapon.DefaultView;

dgWeapon.Columns[0].Visibility = Visibility.Collapsed;

dgWeapon.Columns[8].Visibility = Visibility.Collapsed;

dgWeapon.Columns[9].Visibility = Visibility.Collapsed;

}

private void ChbdgKorz\_Checked(object sender, RoutedEventArgs e)

{

dgWeapon.Visibility = Visibility.Visible;

}

private void ChbdgKorz\_Unchecked(object sender, RoutedEventArgs e)

{

dgWeapon.Visibility = Visibility.Hidden;

}

}

}

Registration.xaml.cs

using WeaponStore.TableConnection;

using System;

using System.Collections.Generic;

using System.Collections.ObjectModel;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace WeaponStore

{

/// <summary>

/// Логика взаимодействия для Registration.xaml

/// </summary>

///

public partial class Registration : Window

{

private TableConnection.ConnectionLicense connectionLicense;

List<TableConnection.ConnectionLicense> Licenses;

async Task<bool> Symb(string str)

{

bool znach = false;

await Task.Run(() =>

{

if (

str.Contains("?") || str.Contains("!") || str.Contains("@") ||

str.Contains("#") || str.Contains("№") || str.Contains("~") ||

str.Contains(";") || str.Contains("%") || str.Contains("$") ||

str.Contains("^") || str.Contains("&") || str.Contains(":") ||

str.Contains("\*") || str.Contains("(") || str.Contains(")") ||

str.Contains("\_") || str.Contains("=") || str.Contains("+") ||

str.Contains("/") || str.Contains("|") || str.Contains("[") ||

str.Contains("]") || str.Contains("{") || str.Contains("}") ||

str.Contains("<") || str.Contains(">") || str.Contains("-") ||

str.Contains(",") || str.Contains("`") || str.Contains("."))

znach = true;

});

return znach;

}

List<TableConnection.ConnectionLicense> licenses;

bool isAdmin { get; set; }

private ConnectionKlient connectionKlient;

public Registration (ConnectionKlient connectionKlient, bool isConnection = false)

{

InitializeComponent();

this.connectionKlient = connectionKlient;

this.isAdmin = isAdmin;

licenses = new DBProcedures().getLicenseList();

ObservableCollection<string> listLicense = new ObservableCollection<string>();

foreach (var licence in licenses)

{

listLicense.Add(licence.License\_Number);

}

this.comboBoxID\_Licence.ItemsSource = listLicense;

if (this.connectionKlient != null)

{

int idlistLicence = -1;

for (idlistLicence = 0; idlistLicence < licenses.Count; idlistLicence++)

{

if (licenses[idlistLicence].ID\_License == this.connectionKlient.License\_ID)

{

break;

}

}

this.comboBoxID\_Licence.ItemsSource = listLicense;

txtBoxSurname.Text = connectionKlient.First\_Name\_Klient;

txtBoxName.Text = connectionKlient.Name\_Klient;

txtBoxMiddle\_Name.Text = connectionKlient.Middle\_Name\_Klient;

txtBoxTelephone\_Number.Text = connectionKlient.Phone\_Number;

//txtBoxLicense.Text = Convert.ToString (connectionKlient.License\_ID);

txtBoxLogin.Text = Login (connectionKlient.ID\_Authorization);

txtBoxPassword.Visibility = Visibility.Hidden;

}

else

{

txtBoxSurname.Text = string.Empty;

txtBoxName.Text = string.Empty;

txtBoxMiddle\_Name.Text = string.Empty;

txtBoxTelephone\_Number.Text = string.Empty;

txtBoxLicense.Text = string.Empty;

txtBoxLogin.Text = string.Empty;

//txtBoxPassword.Visibility = Visibility.Hidden;

}

if (isAdmin)

{

txtBoxSurname.IsReadOnly = true;

txtBoxName.IsReadOnly = true;

txtBoxMiddle\_Name.IsReadOnly = true;

txtBoxTelephone\_Number.IsReadOnly = true;

comboBoxID\_Licence.IsReadOnly = true;

//txtBoxLicense.IsReadOnly = true;

txtBoxLogin.IsReadOnly = true;

txtBoxPassword.Visibility = Visibility.Hidden;

txtBoxPassword2.Visibility = Visibility.Hidden;

btRegistration.Content = "Close";

}

string Login(int ID\_Authorization)

{

var dbprocedures = new DBProcedures();

var auths = dbprocedures.getAuthorizationList();

foreach (var a in auths)

{

if (a.ID\_Authorization == ID\_Authorization)

{

return a.Login;

}

}

return string.Empty;

}

}

int idAuth(string Login)

{

var dbprocedures = new DBProcedures();

var auths = dbprocedures.getAuthorizationList();

foreach (var a in auths)

{

if (a.Login == txtBoxLogin.Text)

{

return a.ID\_Authorization;

}

}

return -1;

}

int idLicence(string Licence)

{

var dbprocedures = new DBProcedures();

var auths = dbprocedures.getLicenseList();

foreach (var a in auths)

{

if (a.License\_Number == txtBoxLogin.Text)

{

return a.ID\_License;

}

}

return -1;

}

private async void BtRegistration\_Click(object sender, RoutedEventArgs e)

{

if (isAdmin)

{

Close();

return;

}

bool znach2 = await Symb(txtBoxSurname.Text);

if (txtBoxSurname.Text == String.Empty || znach2)

{

MessageBox.Show("Есть ошибка в указании Фамилии");

return;

}

bool znach1 = await Symb(txtBoxName.Text);

if (txtBoxName.Text == String.Empty || znach1)

{

MessageBox.Show("Есть ошибка в указании Имени");

return;

}

bool znach3 = await Symb(txtBoxMiddle\_Name.Text);

if (txtBoxMiddle\_Name.Text == String.Empty || znach3)

{

MessageBox.Show("Есть ошибка в указании Отчества");

return;

}

bool znach4 = await Symb(txtBoxTelephone\_Number.Text);

if (txtBoxTelephone\_Number.Text == String.Empty || znach4)

{

MessageBox.Show("Есть ошибка в указании телефонного номера");

return;

}

if (comboBoxID\_Licence.SelectedIndex < 0)

{

MessageBox.Show("Не выбрана категория");

return;

}

if (txtBoxLogin.Text == String.Empty)

{

MessageBox.Show("Не указан логин");

return;

}

if (txtBoxPassword .Text == String.Empty)

{

MessageBox.Show("Не указан пароль");

return;

}

if (txtBoxPassword.Text != txtBoxPassword2.Text)

{

MessageBox.Show("Введенные пароли не совпадают");

return;

}

if (idAuth(txtBoxLogin.Text) >= 0)

{

MessageBox.Show("Такой логин уже есть в базе");

return;

}

if (ch1.IsChecked == false)

{

MessageBox.Show("Проверка на робота не пройдена");

return;

}

int ID\_License = licenses[comboBoxID\_Licence.SelectedIndex].ID\_License;

//int ID\_License = idLicence(txtBoxLicense.Text);

TableConnection.ConnectionAuthorization newAuthorization = new TableConnection.ConnectionAuthorization(-1, txtBoxLogin.Text, txtBoxPassword.Text, 2);

new DBProcedures().spAuthorization\_insert(newAuthorization);

int id\_Auth = idAuth(txtBoxLogin.Text);

TableConnection.ConnectionKlient klient = new TableConnection.ConnectionKlient(

id\_Auth,

txtBoxSurname.Text,

txtBoxName.Text,

txtBoxMiddle\_Name.Text,

txtBoxTelephone\_Number.Text,

ID\_License

);

new DBProcedures().spKlient\_insert(klient);

MessageBox.Show("Регистрация прошла успешно");

Close();

}

private void Button\_Click(object sender, RoutedEventArgs e)

{

LicenceSogl licenceSogl = new LicenceSogl();

licenceSogl.Show();

}

private void BtInsert\_Click(object sender, RoutedEventArgs e)

{

if (txtBoxLicense.Text == string.Empty)

{

MessageBox.Show("Ошибка");

return;

}

new DBProcedures().spLicense\_insert(new TableConnection.ConnectionLicense(

-1,

txtBoxLicense.Text

));

Licenses = (new DBProcedures()).getLicenseList();

comboBoxID\_Licence.ItemsSource = Licenses;

comboBoxID\_Licence.SelectedValuePath = "ID\_License";

comboBoxID\_Licence.DisplayMemberPath = "License\_Number";

}

private void Window\_Activated(object sender, EventArgs e)

{

}

}

}

Shotguns.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace WeaponStore

{

/// <summary>

/// Логика взаимодействия для Shotguns.xaml

/// </summary>

public partial class Shotguns : Window

{

public Shotguns()

{

InitializeComponent();

}

private void BtEVA\_8\_Click(object sender, RoutedEventArgs e)

{

PreyscourantWeaponTest preyscourantWeaponTest = new PreyscourantWeaponTest();

preyscourantWeaponTest.Show();

Visibility = Visibility.Collapsed;

}

private void BtMastiff\_Click(object sender, RoutedEventArgs e)

{

PreyscourantWeaponTest preyscourantWeaponTest = new PreyscourantWeaponTest();

preyscourantWeaponTest.Show();

Visibility = Visibility.Collapsed;

}

private void BtPeacekeeper\_Click(object sender, RoutedEventArgs e)

{

PreyscourantWeaponTest preyscourantWeaponTest = new PreyscourantWeaponTest();

preyscourantWeaponTest.Show();

Visibility = Visibility.Collapsed;

}

private void BtMozambique\_Click(object sender, RoutedEventArgs e)

{

PreyscourantWeaponTest preyscourantWeaponTest = new PreyscourantWeaponTest();

preyscourantWeaponTest.Show();

Visibility = Visibility.Collapsed;

}

}

}

SMG.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace WeaponStore

{

/// <summary>

/// Логика взаимодействия для SMG.xaml

/// </summary>

public partial class SMG : Window

{

public SMG()

{

InitializeComponent();

}

private void BtProwler\_Click(object sender, RoutedEventArgs e)

{

PreyscourantWeaponTest preyscourantWeaponTest = new PreyscourantWeaponTest();

preyscourantWeaponTest.Show();

Visibility = Visibility.Collapsed;

}

private void BtR\_99\_Click(object sender, RoutedEventArgs e)

{

PreyscourantWeaponTest preyscourantWeaponTest = new PreyscourantWeaponTest();

preyscourantWeaponTest.Show();

Visibility = Visibility.Collapsed;

}

private void BtAlternator\_Click(object sender, RoutedEventArgs e)

{

PreyscourantWeaponTest preyscourantWeaponTest = new PreyscourantWeaponTest();

preyscourantWeaponTest.Show();

Visibility = Visibility.Collapsed;

}

}

}

SniperRifles.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace WeaponStore

{

/// <summary>

/// Логика взаимодействия для SniperRifles.xaml

/// </summary>

public partial class SniperRifles : Window

{

public SniperRifles()

{

InitializeComponent();

}

private void BtScout\_G7\_Click(object sender, RoutedEventArgs e)

{

PreyscourantWeaponTest preyscourantWeaponTest = new PreyscourantWeaponTest();

preyscourantWeaponTest.Show();

Visibility = Visibility.Collapsed;

}

private void BtTriple\_Take\_Click(object sender, RoutedEventArgs e)

{

PreyscourantWeaponTest preyscourantWeaponTest = new PreyscourantWeaponTest();

preyscourantWeaponTest.Show();

Visibility = Visibility.Collapsed;

}

private void BtLongbow\_Click(object sender, RoutedEventArgs e)

{

PreyscourantWeaponTest preyscourantWeaponTest = new PreyscourantWeaponTest();

preyscourantWeaponTest.Show();

Visibility = Visibility.Collapsed;

}

private void BtKraber\_Click(object sender, RoutedEventArgs e)

{

PreyscourantWeaponTest preyscourantWeaponTest = new PreyscourantWeaponTest();

preyscourantWeaponTest.Show();

Visibility = Visibility.Collapsed;

}

}

}

Spravochnik.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace WeaponStore

{

/// <summary>

/// Логика взаимодействия для Spravochnik.xaml

/// </summary>

public partial class Spravochnik : Window

{

public Spravochnik()

{

InitializeComponent();

}

private void BtWeapon\_Click(object sender, RoutedEventArgs e)

{

Weapon weapon = new Weapon();

weapon.Show();

Visibility = Visibility.Collapsed;

}

private void BtModifications\_Click(object sender, RoutedEventArgs e)

{

Modifications modifications = new Modifications();

modifications.Show();

Visibility = Visibility.Collapsed; ;

}

private void BtAmmo\_Click(object sender, RoutedEventArgs e)

{

Ammo ammo = new Ammo();

ammo.Show();

Visibility = Visibility.Collapsed;

}

private void BtKlients\_Click(object sender, RoutedEventArgs e)

{

Klients klients = new Klients();

klients.Show();

Visibility = Visibility.Collapsed;

}

private void BtBlockKlient\_Click(object sender, RoutedEventArgs e)

{

BlockKlient blockKlient = new BlockKlient();

blockKlient.Show();

Visibility = Visibility.Collapsed;

}

private void BtSupply\_Click(object sender, RoutedEventArgs e)

{

Supply supply = new Supply();

supply.Show();

Visibility = Visibility.Collapsed;

}

private void BtEmployee\_Click(object sender, RoutedEventArgs e)

{

Employee employee = new Employee();

employee.Show();

Visibility = Visibility.Collapsed;

}

private void BtBack\_Click(object sender, RoutedEventArgs e)

{

//Authorization authorization = new Authorization();

//authorization.Show();

//Visibility = Visibility.Collapsed;

Close();

}

private void BtNakladnaya\_Click(object sender, RoutedEventArgs e)

{

Nakladnaya nakladnaya = new Nakladnaya();

nakladnaya.Show();

Visibility = Visibility.Collapsed;

}

}

}

Supply.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

using System.Data;

using System.Data.SqlClient;

using WeaponStore.TableConnection;

namespace WeaponStore

{

/// <summary>

/// Логика взаимодействия для Supply.xaml

/// </summary>

public partial class Supply : Window

{

private TableConnection.ConnectionSupply1 connectionSupply1;

List<TableConnection.ConnectionSupply1> Supplies;

private TableConnection.ConnectionWeapon connectionWeapon;

List<TableConnection.ConnectionWeapon> Weapon;

private TableConnection.ConnectionAmmo connectionAmmo;

private TableConnection.ConnectionModifications connectionModifications;

private TableConnection.ConnectionDogovor connectionDogovor;

List<TableConnection.ConnectionDogovor> Dogovors;

private TableConnection.ConnectionSupplier connectionSupplier;

List<TableConnection.ConnectionSupplier> Suppliers;

public string extension = string.Empty;

private string QR = "";

public Supply()

{

InitializeComponent();

}

DBProcedures procedures = new DBProcedures();

private void Window\_Activated(object sender, EventArgs e)

{

//Supplies = (new DBProcedures()).getSupplyList();

//dgSupply.ItemsSource = Supplies;

//Dogovors = (new DBProcedures()).getDogovorList();

//lbDogovor.ItemsSource = Dogovors;

//Suppliers = (new DBProcedures()).getSupplierList();

//lbSupplier.ItemsSource = Suppliers;

}

private void dgFill(string qr)

{

ConnectionSupply connection = new ConnectionSupply();

ConnectionSupply.qrSupply = qr;

connection.Supply\_Fill();

dgSupply.ItemsSource = connection.dtSupply.DefaultView;

dgSupply.Columns[0].Visibility = Visibility.Collapsed;

dgSupply.Columns[5].Visibility = Visibility.Collapsed;

dgSupply.Columns[6].Visibility = Visibility.Collapsed;

dgSupply.Columns[8].Visibility = Visibility.Collapsed;

dgSupply.Columns[9].Visibility = Visibility.Collapsed;

}

private void BtInsert\_Click(object sender, RoutedEventArgs e)

{

//new DBProcedures().spSupply\_insert(new TableConnection.ConnectionSupply(

// -1,

// tbDate.Text,

// Convert.ToInt32(tbAmmount\_Accepted\_Weapon.Text),

// Convert.ToInt32(tbAmmount\_Accepted\_Modifications.Text),

// Convert.ToInt32(tbAmmount\_Accepted\_Ammo.Text),

// Convert.ToInt32(lbSupplier.SelectedValue),

// Convert.ToInt32(lbDogovor.SelectedValue)

// ));

//Supplies= (new DBProcedures()).getSupplyList();

//dgSupply.ItemsSource = Supplies;

procedures.spSupply\_insert(tbDate.Text, Convert.ToInt32(tbAmmount\_Accepted\_Weapon.Text), Convert.ToInt32(tbAmmount\_Accepted\_Modifications.Text), Convert.ToInt32(tbAmmount\_Accepted\_Ammo.Text), Convert.ToInt32(lbSupplier.SelectedValue), Convert.ToInt32(lbDogovor.SelectedValue));

dgFill(QR);

}

private void BtUpdate\_Click(object sender, RoutedEventArgs e)

{

//new DBProcedures().spSupply\_update(new TableConnection.ConnectionSupply(

// this.connectionSupply.ID\_Supply,

// tbDate.Text,

// Convert.ToInt32(tbAmmount\_Accepted\_Weapon.Text),

// Convert.ToInt32(tbAmmount\_Accepted\_Modifications.Text),

// Convert.ToInt32(tbAmmount\_Accepted\_Ammo.Text),

// Convert.ToInt32(lbSupplier.SelectedValue),

// Convert.ToInt32(lbDogovor.SelectedValue)

// ));

//Supplies = (new DBProcedures()).getSupplyList();

//dgSupply.ItemsSource = Supplies;

try

{

DataRowView ID = (DataRowView)dgSupply.SelectedItems[0];

if (ID == null)

MessageBox.Show("", "");

else

procedures.spSupply\_update(tbDate.Text, Convert.ToInt32(ID["ID\_Supply"]),Convert.ToInt32(tbAmmount\_Accepted\_Weapon.Text), Convert.ToInt32(tbAmmount\_Accepted\_Modifications.Text), Convert.ToInt32(tbAmmount\_Accepted\_Ammo.Text), Convert.ToInt32(lbSupplier.SelectedValue), Convert.ToInt32(lbDogovor.SelectedValue));

dgFill(QR);

}

catch

{

}

}

private void BtDelete\_Click(object sender, RoutedEventArgs e)

{

//Supplies = (new DBProcedures()).getSupplyList();

//int selectedIndex = dgSupply.SelectedIndex;

//var ap = new DBProcedures().getSupplyList();

//foreach (var it in ap)

//{

// if (it.ID\_Supply == Supplies[selectedIndex].ID\_Supply);

// {

// connectionSupply1 = it;

// break;

// }

//}

//var ap1 = new DBProcedures().getWeaponList();

//foreach (var it in ap1)

//{

// if (it.ID\_Weapon == Weapon[Convert.ToInt32("Ammount\_Weapon")].ID\_Weapon);

// {

// connectionWeapon = it;

// break;

// }

//}

//int Amm\_weapon = connectionSupply1.Ammount\_Accepted\_Weapon;

//int Amm\_ammo = connectionSupply1.Ammount\_Accepted\_Ammo;

//int Amm\_modifications = connectionSupply1.Ammount\_Accepted\_Modifications;

switch (MessageBox.Show("Разгрузить поставку?",

"Разгрузка поставки", MessageBoxButton.YesNo,

MessageBoxImage.Warning))

{

case MessageBoxResult.Yes:

//connectionWeapon.Ammount\_Weapon = connectionWeapon.Ammount\_Weapon + Amm\_weapon;

//connectionAmmo.Ammount\_Ammo =connectionAmmo.Ammount\_Ammo + Amm\_ammo;

//connectionModifications.Ammount\_Modifications = connectionModifications.Ammount\_Modifications + Amm\_modifications;

DataRowView ID =

(DataRowView)dgSupply.SelectedItems[0];

procedures.spSupply\_delete(

Convert.ToInt32(ID["ID\_Supply"]));

dgFill(QR);

break;

}

}

private void BtInsertSupplier\_Click(object sender, RoutedEventArgs e)

{

new DBProcedures().spSupplier\_insert(new TableConnection.ConnectionSupplier(

-1,

tbName\_Organization.Text

));

Suppliers = (new DBProcedures()).getSupplierList();

lbSupplier.ItemsSource = Suppliers;

}

private void BtUpdateSupplier\_Click(object sender, RoutedEventArgs e)

{

new DBProcedures().spSupplier\_update(new TableConnection.ConnectionSupplier(

this.connectionSupplier.ID\_Supplier,

tbName\_Organization.Text

));

Suppliers = (new DBProcedures()).getSupplierList();

lbSupplier.ItemsSource = Suppliers;

}

private void BtDeleteSupplier\_Click(object sender, RoutedEventArgs e)

{

switch (MessageBox.Show("Удалить выбранную запись?",

"Удаление записи", MessageBoxButton.YesNo,

MessageBoxImage.Warning))

{

case MessageBoxResult.Yes:

procedures.spSupplier\_delete(

Convert.ToInt32(lbSupplier.

SelectedValue.ToString()));

Suppliers = (new DBProcedures()).getSupplierList();

lbSupplier.ItemsSource = Suppliers;

break;

}

}

private void BtInsertDogovor\_Click(object sender, RoutedEventArgs e)

{

new DBProcedures().spDogovor\_insert(new TableConnection.ConnectionDogovor(

-1,

tbNumber\_Dogovor.Text

));

Dogovors = (new DBProcedures()).getDogovorList();

lbDogovor.ItemsSource = Dogovors;

}

private void BtUpdateDogovor\_Click(object sender, RoutedEventArgs e)

{

new DBProcedures().spDogovor\_update(new TableConnection.ConnectionDogovor(

this.connectionDogovor.ID\_Dogovor,

tbNumber\_Dogovor.Text

));

Dogovors = (new DBProcedures()).getDogovorList();

lbDogovor.ItemsSource = Dogovors;

}

private void BtDeleteDogovor\_Click(object sender, RoutedEventArgs e)

{

switch (MessageBox.Show("Удалить выбранную запись?",

"Удаление записи", MessageBoxButton.YesNo,

MessageBoxImage.Warning))

{

case MessageBoxResult.Yes:

procedures.spEmployee\_delete(

Convert.ToInt32(lbDogovor.

SelectedValue.ToString()));

Dogovors = (new DBProcedures()).getDogovorList();

lbDogovor.ItemsSource = Dogovors;

break;

}

}

private void BtSearch\_Click(object sender, RoutedEventArgs e)

{

foreach (DataRowView dataRow in (DataView)dgSupply.ItemsSource)

{

if (dataRow.Row.ItemArray[1].ToString() == tbSearch.Text ||

dataRow.Row.ItemArray[2].ToString() == tbSearch.Text ||

dataRow.Row.ItemArray[3].ToString() == tbSearch.Text ||

dataRow.Row.ItemArray[4].ToString() == tbSearch.Text ||

dataRow.Row.ItemArray[7].ToString() == tbSearch.Text ||

dataRow.Row.ItemArray[10].ToString() == tbSearch.Text)

{

dgSupply.SelectedItem = dataRow;

}

}

}

private void BtFilter\_Click(object sender, RoutedEventArgs e)

{

switch (chbFilter.IsChecked)

{

case (true):

string newQR = QR + " where [Date] like '%" + tbSearch.Text + "%' or " +

"[Ammount\_Accepted\_Weapon] like '%" + tbSearch.Text + "%' or " +

"[Ammount\_Accepted\_Modifications] like '%" + tbSearch.Text + "%' or " +

"[Ammount\_Accepted\_Ammo] like '%" + tbSearch.Text + "%' or " +

"[Supplier\_ID] like '%" + tbSearch.Text + "%' or " +

"[Dogovor\_ID] like '%" + tbSearch.Text + "%'";

dgFill(newQR);

break;

case (false):

dgFill(QR);

break;

}

}

private void Window\_Loaded(object sender, RoutedEventArgs e)

{

dgSupply.Visibility = Visibility.Hidden;

QR = ConnectionSupply.qrSupply;

dgFill(QR);

Dogovors = (new DBProcedures()).getDogovorList();

lbDogovor.ItemsSource = Dogovors;

lbDogovor.SelectedValuePath = "ID\_Dogovor";

lbDogovor.DisplayMemberPath = "Number\_Dogovor";

Suppliers = (new DBProcedures()).getSupplierList();

lbSupplier.ItemsSource = Suppliers;

lbSupplier.SelectedValuePath = "ID\_Supplier";

lbSupplier.DisplayMemberPath = "Name\_Organization";

}

private void BtBack\_Click(object sender, RoutedEventArgs e)

{

Spravochnik spravochnik = new Spravochnik();

spravochnik.Show();

Visibility = Visibility.Collapsed;

}

private void ChbdgSupply\_Checked(object sender, RoutedEventArgs e)

{

dgSupply.Visibility = Visibility.Visible;

dgFill(QR);

}

private void ChbdgSupply\_Unchecked(object sender, RoutedEventArgs e)

{

dgSupply.Visibility = Visibility.Hidden;

dgFill(QR);

}

}

}

Vakansii.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace WeaponStore

{

/// <summary>

/// Логика взаимодействия для Vakansii.xaml

/// </summary>

public partial class Vakansii : Window

{

public Vakansii()

{

InitializeComponent();

}

private void Button\_Click(object sender, RoutedEventArgs e)

{

MainWindow spravochnik = new MainWindow();

spravochnik.Show();

Visibility = Visibility.Collapsed;

}

}

}

Weapon.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

using System.Data;

using System.Data.SqlClient;

using WeaponStore.TableConnection;

namespace WeaponStore

{

/// <summary>

/// Логика взаимодействия для Weapon.xaml

/// </summary>

///

public partial class Weapon : Window

{

private TableConnection.ConnectionType\_Weapon connectionType\_Weapon;

private TableConnection.ConnectionWeapon connectionWeapon;

List<TableConnection.ConnectionWeapon> Weapons;

List<TableConnection.ConnectionType\_Weapon> Type\_Weapons;

List<TableConnection.ConnectionWeaponDetail> ListWeapons;

public string extension = string.Empty;

private string QR = "";

public Weapon()

{

InitializeComponent();

}

private void Window\_Activated(object sender, EventArgs e)

{

//dgWeapon.Visibility = Visibility.Hidden;

//lbTypeWeapons.Visibility = Visibility.Hidden;

//ListWeapons = (new DBProcedures()).getWeaponListDetail();

//dgWeapon.ItemsSource = ListWeapons;

//dgWeapon.Columns[0].Visibility = Visibility.Collapsed;

//Type\_Weapons = (new DBProcedures()).getType\_WeaponList();

//lbTypeWeapons.ItemsSource = Type\_Weapons;

//lbTypeWeapons.SelectedValuePath = "ID\_Type\_Weapon";

//lbTypeWeapons.DisplayMemberPath = "Name\_Type\_Weapon";

}

DBProcedures procedures = new DBProcedures();

private void BtInsert\_Click(object sender, RoutedEventArgs e)

{

if (tbNameWeapon.Text == string.Empty ||

tbAccuracy.Text == string.Empty ||

tbFireRate.Text == string.Empty ||

tbShellsInStore.Text == string.Empty ||

tbAmmount\_Weapon.Text == string.Empty ||

tbCost.Text == string.Empty ||

lbTypeWeapons.SelectedIndex < 0)

{

MessageBox.Show("Ошибка");

return;

}

new DBProcedures().spWeapon\_insert(new TableConnection.ConnectionWeapon(

-1,

tbNameWeapon.Text,

tbAccuracy.Text,

tbFireRate.Text,

tbShellsInStore.Text,

Convert.ToInt32(tbAmmount\_Weapon.Text),

tbCost.Text,

Convert.ToInt32(lbTypeWeapons.SelectedValue)

)) ;

ListWeapons = (new DBProcedures()).getWeaponListDetail();

dgWeapon.ItemsSource = ListWeapons;

dgWeapon.Columns[0].Visibility = Visibility.Collapsed;

}

private void BtDelete\_Click(object sender, RoutedEventArgs e)

{

int selectedIndex = dgWeapon.SelectedIndex;

new DBProcedures().spWeapon\_delete(ListWeapons[selectedIndex].ID\_Weapon);

MessageBox.Show("Операция выполнена");

ListWeapons = (new DBProcedures()).getWeaponListDetail();

dgWeapon.ItemsSource = ListWeapons;

dgWeapon.Columns[0].Visibility = Visibility.Collapsed;

}

private void BtUpdate\_Click(object sender, RoutedEventArgs e)

{

int selectedIndex = dgWeapon.SelectedIndex;

var ap = new DBProcedures().getWeaponList();

foreach (var it in ap)

{

if (it.ID\_Weapon == Weapons[selectedIndex].ID\_Weapon);

{

connectionWeapon = it;

break;

}

}

if (tbNameWeapon.Text == string.Empty ||

tbAccuracy.Text == string.Empty ||

tbFireRate.Text == string.Empty ||

tbShellsInStore.Text == string.Empty ||

tbAmmount\_Weapon.Text == string.Empty ||

tbCost.Text == string.Empty ||

lbTypeWeapons.SelectedIndex < 0)

{

MessageBox.Show("Ошибка");

return;

}

new DBProcedures().spWeapon\_update(new TableConnection.ConnectionWeapon(

this.connectionWeapon.ID\_Weapon,

tbNameWeapon.Text,

tbAccuracy.Text,

tbFireRate.Text,

tbShellsInStore.Text,

Convert.ToInt32(tbAmmount\_Weapon.Text),

tbCost.Text,

Convert.ToInt32(lbTypeWeapons.SelectedValue)

));

ListWeapons = (new DBProcedures()).getWeaponListDetail();

dgWeapon.ItemsSource = ListWeapons;

dgWeapon.Columns[0].Visibility = Visibility.Collapsed;

}

private void BtInsertTypeWeapon\_Click(object sender, RoutedEventArgs e)

{

if (tbName\_Type\_Weapon.Text == string.Empty)

{

MessageBox.Show("Ошибка");

return;

}

new DBProcedures().spType\_Weapon\_insert(new TableConnection.ConnectionType\_Weapon(

-1,

tbName\_Type\_Weapon.Text

));

Type\_Weapons = (new DBProcedures()).getType\_WeaponList();

lbTypeWeapons.ItemsSource = Type\_Weapons;

lbTypeWeapons.SelectedValuePath = "ID\_Type\_Weapon";

lbTypeWeapons.DisplayMemberPath = "Name\_Type\_Weapon";

}

private void BtUpdateTypeWeapon\_Click(object sender, RoutedEventArgs e)

{

int selectedIndex = lbTypeWeapons.SelectedIndex;

var ap = new DBProcedures().getType\_WeaponList();

foreach (var it in ap)

{

if (it.ID\_Type\_Weapon == Type\_Weapons[selectedIndex].ID\_Type\_Weapon) ;

{

connectionType\_Weapon = it;

break;

}

}

if (tbName\_Type\_Weapon.Text == string.Empty)

{

MessageBox.Show("Ошибка");

return;

}

new DBProcedures().spType\_Weapon\_update(new TableConnection.ConnectionType\_Weapon(

this.connectionType\_Weapon.ID\_Type\_Weapon,

tbName\_Type\_Weapon.Text

));

Type\_Weapons = (new DBProcedures()).getType\_WeaponList();

lbTypeWeapons.ItemsSource = Type\_Weapons;

lbTypeWeapons.SelectedValuePath = "ID\_Type\_Weapon";

lbTypeWeapons.DisplayMemberPath = "Name\_Type\_Weapon";

}

private void BtDeleteTypeWeapon\_Click(object sender, RoutedEventArgs e)

{

int selectedIndex = lbTypeWeapons.SelectedIndex;

new DBProcedures().spType\_Weapon\_delete(Type\_Weapons[selectedIndex].ID\_Type\_Weapon);

MessageBox.Show("Операция выполнена");

Type\_Weapons = (new DBProcedures()).getType\_WeaponList();

lbTypeWeapons.ItemsSource = Type\_Weapons;

}

private void BtSearch\_Click(object sender, RoutedEventArgs e)

{

foreach (DataRowView dataRow in (DataView)dgWeapon.ItemsSource)

{

if (dataRow.Row.ItemArray[1].ToString() == tbSearch.Text ||

dataRow.Row.ItemArray[2].ToString() == tbSearch.Text ||

dataRow.Row.ItemArray[3].ToString() == tbSearch.Text ||

dataRow.Row.ItemArray[4].ToString() == tbSearch.Text ||

dataRow.Row.ItemArray[5].ToString() == tbSearch.Text ||

dataRow.Row.ItemArray[6].ToString() == tbSearch.Text)

{

dgWeapon.SelectedItem = dataRow;

}

}

}

private void BtFilter\_Click(object sender, RoutedEventArgs e)

{

string newQR = QR + " where [Name\_Weapon] like '%" + tbSearch.Text + "%' or " +

"[Accuracy] like '%" + tbSearch.Text + "%' or " +

"[Fire\_Rate] like '%" + tbSearch.Text + "%'or " +

"[Shells\_In\_Store] like '%" + tbSearch.Text + "%'or " +

"[Ammount\_Weapon] like '%" + tbSearch.Text + "%'or " +

"[Cost] like '%" + tbSearch.Text + "%'or " +

"[Type\_Weapon\_ID] like '%" + tbSearch.Text + "%'";

ListWeapons = (new DBProcedures()).getWeaponListDetail();

dgWeapon.ItemsSource = ListWeapons;

}

private void DgWeapon\_AutoGeneratingColumn(object sender, DataGridAutoGeneratingColumnEventArgs e)

{

//switch (e.Column.Header)

//{

// case ("Name\_Weapon"):

// e.Column.Header = "Фамилия заключённого";

// break;

// case ("Name\_Prisoner"):

// e.Column.Header = "Имя заколючённого";

// break;

// case ("MiddleName\_Prisoner"):

// e.Column.Header = "Отчество заключённого";

// break;

// case ("Name\_of\_block"):

// e.Column.Header = "Название блока";

// break;

// case ("Surname\_Guardian"):

// e.Column.Header = "Фамилия охранника";

// break;

// case ("Name\_Guardian"):

// e.Column.Header = "Имя охранника";

// break;

// case ("MiddleName\_Guardian"):

// e.Column.Header = "Отчество охранника";

// break;

//}

}

private void BtClose\_Click(object sender, RoutedEventArgs e)

{

Spravochnik spravochnik = new Spravochnik();

spravochnik.Show();

Visibility = Visibility.Collapsed;

}

private void ChdgWeapon\_Checked(object sender, RoutedEventArgs e)

{

dgWeapon.Visibility = Visibility.Visible;

lbTypeWeapons.Visibility = Visibility.Visible;

}

private void ChdgWeapon\_Unchecked(object sender, RoutedEventArgs e)

{

dgWeapon.Visibility = Visibility.Hidden;

lbTypeWeapons.Visibility = Visibility.Hidden;

}

MediaPlayer player = new MediaPlayer();

private void Window\_Loaded(object sender, RoutedEventArgs e)

{

player.Open(new Uri("C:/Users/1/Desktop/Новый семестр-новые страдания/Курсовой/Интерфейс1/Fortnite Default Dance Bass Boosted.mp3", System.UriKind.RelativeOrAbsolute));

player.Play();

dgWeapon.Visibility = Visibility.Hidden;

lbTypeWeapons.Visibility = Visibility.Hidden;

ListWeapons = (new DBProcedures()).getWeaponListDetail();

dgWeapon.ItemsSource = ListWeapons;

dgWeapon.Columns[0].Visibility = Visibility.Collapsed;

Type\_Weapons = (new DBProcedures()).getType\_WeaponList();

lbTypeWeapons.ItemsSource = Type\_Weapons;

lbTypeWeapons.SelectedValuePath = "ID\_Type\_Weapon";

lbTypeWeapons.DisplayMemberPath = "Name\_Type\_Weapon";

}

private void Button\_Click(object sender, RoutedEventArgs e)

{

player.Open(new Uri("C:/Users/1/Desktop/Новый семестр-новые страдания/Курсовой/Интерфейс1/Fortnite Default Dance Bass Boosted.mp3", System.UriKind.RelativeOrAbsolute));

player.Play();

}

}

}

Zastavka.xaml.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace WeaponStore.TableConnection

{

/// <summary>

/// Логика взаимодействия для Zastavka.xaml

/// </summary>

public partial class Zastavka : Window

{

public Zastavka()

{

InitializeComponent();

}

private void Button\_Click(object sender, RoutedEventArgs e)

{

if (!(new DBProcedures().IsConnection))

{

MessageBox.Show("Ошибка подкл к БД");

}

else

{

new Authorization().Show();

}

Close();

}

private void Button\_Click\_1(object sender, RoutedEventArgs e)

{

Form1 form1 = new Form1();

form1.Show();

Visibility = Visibility.Collapsed;

}

}

}